P. 1877/83

9 1983

informatyka

Sieci teleinformatyczne

Współprogramy

Języki symulacyjne

NATIONAL SEMICONDUCTOR 16000

Jeszcze o ZX81

WYDAWNICTWO NOT SIGMA

Płatne rubryki

Propozycję tę kierujemy do dużych instytucji informatycznych, liczących się na polskim rynku — instytucji usługowych, produkujących sprzęt, oprogramowanie, szkolących kadry itd., do wszystkich, których działalność ma szersze, nie tylko lokalne czy doraźne znaczenie.

Proponujemy im stałą obecność na łamach INFORMATYKI, pod własną winietą i — za pieniądze. W przypadku potentatów (ELWRO, ZETO, PTI, MERA-Błonie, ETOB, uczelnie...) — w każdym (lub prawie każdym) numerze, w innych — co któryś — ukazywałaby się firmowana przez daną instytucję jedna kolumna (strona w piśmie) lub jakaś jej część.

Umowa między nami określałaby liczbę kolumn w roku, jaka miałaby być dla danej instytucji przeznaczona. Opłata za jedną kolumnę wynosi 13,2 tys. zł.

Proponujemy, aby każda decydująca się na współpracę z nami instytucja wybrała przedstawiciela, który reprezentowałby w redakcji interesy swojej firmy. Byłby to więc rodzaj rzecznika prasowego i — jednocześnie — koordynatora wydawnictw. Zapewne bowiem dałoby się pogodzić pracę tej osoby przy przygotowaniu materiałów dla redakcji z innymi jej obowiązkami edytorskimi, a chyba także — z normalnymi, zawodowymi.

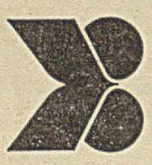
Dzięki takiej innowacji moglibyśmy liczyć na zwiększony przepływ informacji i poglądów. Instytucje same określałyby swój profil wydawniczy (oczywiście, z naszą pomocą), my zaś mielibyśmy gwarancję systematycznego napływu materiałów. I — co ważne — przestalibyśmy przynosić deficyt.

Prosimy o niezwłoczne rozpatrzenie naszej oferty. I szybki kontakt. Pomyślne ustalenia — jak sądzimy — przyniosą korzyści nie tylko partykularne.

REDAKCJA

NACZELNA ORGANIZACJA TECHNICZNA

WYDAWNICTWO



SIGMA

CZASOPISM I KSIĄŻEK TECHNICZNYCH

ul. Świętokrzyska 14a
00-950 Warszawa
skrytka pocztowa 1004

KOLEGIUM REDAKCYJNE

Redaktor naczelny: prof. dr hab. Leon ŁUKASZEWICZ

mgr inż. Zbigniew GLUZA, dr Janusz GWIAZDA, Władysław KLEPACZ (zastępca redaktora naczelnego), dr inż. Tomasz PAWLAK, mgr Marek SOBCZYK, mgr Andrzej SZAŁAS, mgr Zbigniew ŚWIRSKI, dr inż. Janusz ZALEWSKI

Sekretarz redakcji: mgr Teresa JABŁOŃSKA

RADA PROGRAMOWA

Prof. dr hab. Tadeusz PECHE (przewodniczący), mgr inż. Tomasz BAŃKOWSKI (sekretarz), mgr inż. Antoni BOSSOWSKI, mgr inż. Roman BURNO, prof. dr hab. Andrzej JANICKI, mgr inż. Jan KRAMARCZUK, prof. dr hab. inż. Juliusz KULIKOWSKI, prof. dr hab. Leon ŁUKASZEWICZ, prof. dr hab. Antoni MAZURKIEWICZ, gen. dr inż. Marian PASTERNAK, dr inż. Bronisław PIWOWAR, mgr Zbigniew SUBSTYK, prof. dr hab. Tadeusz WALCZAK

Materiałów nie zamówionych Redakcja nie zwraca.

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pok. 243 i 244, tel. 27-71-40, dyżury redakcji 10.00—12.00

Zakł. Graf: „Tamka" Zam. 2235. Obj. 5,0 ark. druk. Nakład 3300 egz. M-106.

Cena egzemplarza zł 150.— INDEKS 36124 Prenumerata roczna zł 900.—

P. 1877/83

Informatyka

zastosowania w gospodarce, technice i nauce

Nr 9
MIESIĘCZNIK
1983
ROK XVIII
Wrzesień

ORGAN KOMITETU INFORMATYKI MINISTERSTWA NAUKI, SZKOLNICTWA WYŻSZEGO I TECHNIKI ORAZ KOMITETU NAUKOWO-TECHNICZNEGO NOT DS. INFORMATYKI

W NUMERZE: Strona

Sieci teleinformatyczne
Czesław Syc, Jerzy Paczocha 2

Programowanie za pomocą współprogramów
Wiesława Bartol 7

Systemy graficzne: percepcja wzrokowa
Jacek Owczarczyk 10

Aspekty użytkowe języków symulacyjnych (1). Symulacja dyskretna
Teresa Wilczek 13

Przegląd mikroprocesorów 16-bitowych (6). NATIONAL SEMICONDUCTOR 16000
Jerzy Grabowski 17

Jeszcze o ZX81
Jerzy Karczmarczuk 22

ALGORYTMY

Algorytmy „kalendarzowe"
Andrzej Szałas, Zbigniew Świrski 25

Z KRAJU

55 Międzynarodowe Targi Poznańskie:

— Rozwój rynku mikrokomputerowego
Jacek Żebrowski 26

— Komputery firm polonijno-zagranicznych
Janusz Zalewski 29

ZE ŚWIATA

Oprogramowanie mikrokomputerów
Oprac. Jacek Walas 32

Mikrokomputer na co dzień
Michał Ziębiński 34

TERMINOLOGIA

Słowniczek terminów związanych z systemami graficznymi
Jacek Owczarczyk 37

W SKRÓCIE 38

POGLĄDY

Rynek dla informatyki
Andrzej Zienkiewicz III okł.

CZESŁAW SYC
JERZY PACZOCHA
Ośrodek Informatyki Technicznej
Warszawa

Sieci teleinformatyczne

Myślą przewodnią tego artykułu jest wykazanie prawdziwości tezy, iż w Polsce jest już obecnie potrzebna — i możliwa do realizacji — sieć teleinformatyczna jako usługa publiczna, podobnie jak np. sieć teleksowa lub telefoniczna. Podjęto również rozważania na temat podstawowej koncepcji takiej sieci. W rozważaniach tych wzięto pod uwagę zarówno potrzeby, jak i możliwości krajowe — obecne oraz w perspektywie 3—5 lat, z uwzględnieniem doświadczeń krajów wysoko rozwiniętych i postępu naukowo-technicznego, zwłaszcza w zakresie zastosowania układów mikroprocesorowych.

W przeszłości przy podejmowaniu decyzji dotyczących realizacji sieci teleinformatycznej nie zawsze brano u nas pod uwagę konieczność spełnienia razem dwóch podstawowych warunków: potrzeby budowy sieci teleinformatycznej oraz możliwości technicznych i ekonomicznych jej realizacji. Często pierwszy z tych warunków nie był w ogóle brany pod uwagę, drugi zaś opierano głównie o technikę zagraniczną, zaś czynniki ekonomiczne, podobnie jak w innych dziedzinach gospodarki, praktycznie nie odgrywały znaczącej roli.

Zapyta ktoś, czy dzisiaj sytuacja się zmieniła. Odpowiedź jest prosta: tak, i to bardzo. Po pierwsze — dziś, jak nigdy dotąd, istnieje realne zapotrzebowanie użytkowników na usługi teleinformatyczne. W tym zakresie z łączy dzierżawionych korzysta już ok. 1000 użytkowników, z telefonicznej sieci komutowanej — ok. 100, a z sieci telegraficznej (dla wolnej transmisji danych) — ok. 50 użytkowników, w tym w ramach systemu BIST zainstalowano ok. 30 stacji umożliwiających jednoczesną współpracę minikomputerów typu MERA 305, MERA 400, DP lub MERA 9150 z ponad 200 łączami komutowanymi sieci teleksowej. Ogólnie w sieci transmisji danych działa w kraju ok. 4000 terminali, w tym ok. 100 terminali inteligentnych oraz ok. 2500 monitorów ekranowych. Zainstalowanych jest 37 procesorów komunikacyjnych, ok. 100 multiplekserów oraz 168 jednostek grupowych.

Należy zauważyć, że współczesne minikomutery i komputery wyposażone są w urządzenia umożliwiające przyłączanie ich do sieci teleinformatycznej. A dla zaspokojenia rosnących obecnie — za sprawą reformy gospodarczej — potrzeb informacyjnych niezbędne staje się stworzenie sieci umożliwiających szerokie świadczenie usług teleinformatycznych.

Polska ma wiele osiągnięć w zakresie badań teoretycznych nad sieciami teleinformatycznymi. Powstały nawet swego rodzaju szkoły, jak np. gdańska — pod kierunkiem prof. Seidlera, wrocławska — pod kierunkiem doc. Bazewicza, warszawska — pod kierunkiem prof. T. Nowickiego i wiele innych. Również nasz ośrodek oraz inne tego typu instytucje — wniosły znaczący wkład do rozwoju tej dziedziny.

W zakresie rozwiązań technicznych mamy już własny mikroprocesor typu INTEL 8080 i kilka innych układów mikroprocesorowych. Chociaż brakuje najważniejszych pamięciowych układów, jak EPROM i RAM (choćby tych, które umożliwiają zestawienie mikrokomputera średniej klasy typu INTEL 8080), to jednak mamy dostęp do podobnych układów w krajach RWPG. W przyszłym roku CEMI być może upora się ze swoimi trudnościami i wyprodukuje wreszcie dawno już zapowiadane układy pamięci 2114 i 2716. Wszystko to rokuje nadzieje, iż w 1984 roku technika mikroprocesorowa rozstrzygnie nie tylko o rodzaju budowanej sieci teleinformatycznej (KŁ lub KP), ale stworzy realne możliwości jej budowy. Zresztą, takie możliwości już w pełni obecnie istnieją.

Z ekonomicznego punktu widzenia podejścia mogą być tu wielorakie:

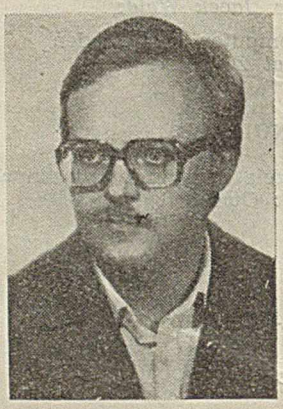
— sieć teleinformatyczna może być budowana sukcesywnie i w sposób modularny, w oparciu o środki przekazane przez użytkowników

— do budowy sieci można powołać spółkę akcyjną, której udziałowcy byliby przyszłymi głównymi jej użytkownikami

— resort łączności w ramach funduszu rozwoju telekomunikacji (FRT) i w oparciu o istniejącą sieć telekomunikacyjną mógłby we własnym zakresie zbudować taką sieć, oferując użytkownikom informatyki różne usługi teleinformatyczne (np. usługi sieci KŁ, KP oraz takie usługi telematyczne, jak VIEWDATA czy TELETEX).

Wszystkie te, a także inne, nie wymienione tu propozycje (np. automatyczne skomutowanie obecnie dzierżawionych łączy w celu ich bardziej efektywnego wykorzystania oraz obniżenia kosztów eksploatacji) można już obecnie rozważać jako przedsięwzięcia ekonomicznie uzasadnione.

Doc. dr inż. CZESŁAW SYC związany jest z informatyką od zakończenia studiów w 1961 r. Kieruje Ośrodkiem Informatyki Technicznej w Warszawie od chwili jego powstania, tj. od 1972 r. Zajmuje się opracowywaniem i realizacją terminali inteligentnych na potrzeby teleinformatyki i telematyki.



Inż. JERZY S. PACZOCHA ukończył studia na Wydziale Elektroniki Politechniki Warszawskiej w 1980 r. Od 1974 r. pracował w Centralnym Ośrodku Badań i Rozwoju Techniki Kolejnictwa, gdzie zajmował się wykorzystaniem minikomputerów w systemach teleinformatycznych i systemach zdalnego sterowania. W Ośrodku Informatyki Technicznej pracuje od 1982 r., gdzie jest kierownikiem Pracowni Systemów Mikroprocesorowych. Prowadzi prace rozwojowe związane z zastosowaniem układów mikroprocesorowych w systemach teleinformatycznych.

Tabela 1. Porównanie cech użytkowych różnych systemów komutacyjnych (wg danych z książki V. Ahuja: Desing Analysis of Computer Communication Network, McGraw-Hill, 1982)

	Wskaźnik	Komutacja łączy	Komutacja wiadomości	Komutacja pakietów
Współpraca między użytkownikami o różnych szybkościach transmisji	+	nie	tak	tak
Lepsze wykorzystanie łączy (ang. Pipeling efect)	+	nie	nie	tak
Konwersja kodów w czasie transmisji	+	nie	tak	tak
Możliwość wyboru szybszej drogi w przypadku zatoru sieci	+	nie	nie	tak
Przesyłanie pakietów wg priorytetów, dodatkowa opłata za pakiet „pilny"	+	nie	nie	tak
Możliwość zabezpieczenia przed błędami	+	możliwa	tak	tak
Buforowanie w węzłach	—	nie	tak	tak
Możliwość ustalenia końca kolejki w przypadku zatoru sieci	—	nie	tak	tak
Procedury kierowania przepływem sieci	—	nie	tak	tak
Ustalanie sekwencji i numerowanie pakietów	—	nie	możliwe	tak
Żądanie segmentacji pakietu i umieszczenie nagłówka	—	nie	tak	tak
Możliwość przesyłania mowy w trakcie transmisji	+	tak	(w trakcie studiów)	

ROZWÓJ SIECI TELEINFORMATYCZNYCH

Na ewolucję i kierunki rozwoju sieci transmisji danych duży wpływ wywarł postęp w dziedzinie elektroniki, zwłaszcza pojawienie się w połowie lat siedemdziesiątych mikroprocesorów. Rozwój sieci obejmował wszystkie podstawowe jej elementy (terminale, układy transmisyjne, urządzenia i techniki komutacji).

Sieci teleinformatyczne powstawały jako zamknięte sieci komputerowe w oparciu o łącza dzierżawione, a także jako systemy otwarte, działające w oparciu o publiczną komutowaną sieć telefoniczną i telegraficzną. Wykorzystanie publicznej telefonicznej sieci komutowanej do celów transmisji danych jest rozwiązaniem kompromisowym, które narzuciło konieczność budowy specjalizowanych sieci transmisji danych.

Udoskonalenie prostych asynchronicznych terminali umożliwia realizację sieci transmisji danych z przestrzenną komutacją kanałów. Sieć taka różni się od teleksowej jedynie większą szybkością transmisji. Dalszy zaś rozwój elektroniki pozwolił na realizację central z komutacją czasową, co umożliwiło rozproszenie sieci oraz realizację sieci zintegrowanej ISDN (Integrated Services Digital Network).

Rozwój techniki komputerowej pozwolił realizować sieci z komutacją wiadomości, oparte na typowych systemach komputerowych i minikomputerowych, a ostatnio mikrokomputerowych, i przyłączonych do nich terminali. Przesyłanie wiadomości było tu oparte np. na wielodostępnej bazie danych. W konsekwencji prowadziło to do powstania sieci komputerowych. Z chwilą pojawienia się układów o dużym stopniu scalenia (LSI) i układów mikroprocesorowych stworzona została możliwość przerzucenia funkcji teletransmisyjnych na specjalizowane procesory komunikacyjne. Doprowadziło to do powstania sieci teleinformatycznych z komutacją pakietów.

Jaka jest więc różnica pomiędzy siecią transmisji danych a siecią teleinformatyczną?

Sieć transmisji danych nie uwzględnia właściwości źródła informacji (np. abonenta), a jej zasadniczym parametrem jest szybkość transmisji. Podstawowymi parametrami sieci teleinformatycznej przenoszącej z reguły informację nieciągłą jest natomiast wielkość strumienia danych, jego wielkość szczytowa i czas przenoszenia porcji informacji (pakietu).

W tabeli 1 przedstawiono porównanie sieci z punktu widzenia jej użytkownika, według kryterium sposobu komutacji. Z tabeli tej wynika, że lepsza organizacja sieci powoduje zwiększenie wymagań ze strony użytkownika. Plusem zaznaczono cechy korzystne, minusem — cechy wprowadzające dodatkowe uwarunkowania. Celowo pominięto nazwy systemów będących kolejnymi etapami rozwoju.

Obecny rozwój sieci teleinformatycznych przebiega w następujących dwóch kierunkach:

- zintegrowanej sieci z komutacją łączy CSDN (Circuit Switched Data Network) lub ISDN (Integrated Services Digital Network) w oparciu o centrale telefoniczne PCM (np. E-10)
- sieci z komutacją pakietów PSDN (Packet Switched Data Network).

Głównymi reprezentantami sieci typu CSDN są: ITT 12, NPDN (Nordic Public Data Network), Caduses. W sieci tego rodzaju abonent dysponuje kanałem o szybkości transmisji 64 KB/s. Po nawiązaniu połączenia abonenci mogą wykorzystywać sieci do różnych celów: rozmów telefonicznych cyfrowych PCM, transmisji danych, telekopii (fascimile) i innych (rys. 1). Wg zaleceń CCITT protokół abonencki powinien być tu zgodny z zaleceniem X.21.

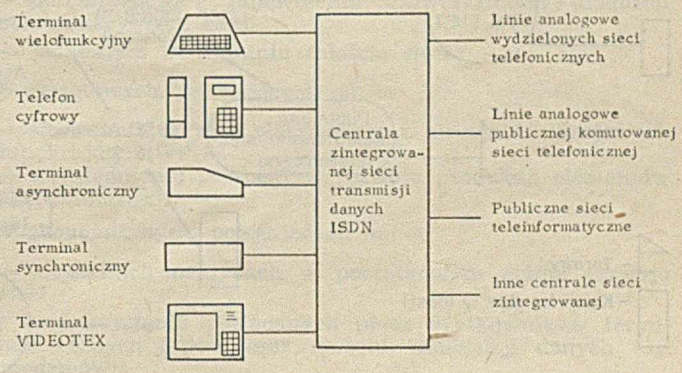
Najbardziej reprezentatywne sieci typu PSDN to: IPSS (International Packet Switching Service), EUROPACK, EDX-P (SIMENS), TYMNET (USA), TRANSPACK (Francja), WUI PSDN (USA). Zalecenia przewidują stosowanie na poziomie abonenckim protokołu X.25 lub innych, natomiast pomiędzy węzłami sieci i sieciami międzynarodowymi — protokołu X.75.

Sieci pakietowe umożliwiają współpracę pomiędzy terminami i komputerami o różnych szybkościach, kodach oraz różnymi sieciami, np. telefoniczną komutowaną i teleksową (rys. 2), co w istotny sposób przyczynia się do ich rozwoju. Zaletami tych sieci są przede wszystkim — lepsze wykorzystanie łączy oraz możliwość rozbudowy (rys. 3).

Oto przykładowe dane obrazujące rozwój tego typu sieci:

- TRANSPACK (Francja): początek 1978 r. — 4 węzły sieci, 1979 r. — 12 węzłów, 1981 r. — 19 węzłów (graniczny czas przesłania pakietu przez sieć 150 ms).
- TYMNET (USA): początek — 1978 r., obecnie — 300 węzłów sieci.

Trudno dzisiaj powiedzieć, który z kierunków będzie dominujący. Obniżenie kosztów transmisji poprzez wprowadzenie łączy światłowodowych przyczyni się prawdopodobnie do rozwoju sieci typu CSDN. Układy o dużym stopniu scalenia (LSI) umożliwiają budowę cyfrowych aparatów telefonicznych i dalsze zwiększenie ich funkcji. Pojawiają się także układy liniowe do transmisji PCM (np. ITT produkuje specjalizowane układy o dużej skali integracji do budowy central elektronicznych dla sieci zintegrowanej). Natomiast sieć z komutacją pakietów PSDN wymaga mniejszych nakładów w początkowym etapie, a możliwość współpracy z różnego rodzaju sieciami i terminalami,



Rys. System ITT12 jako przykład integrowanej usługowej sieci transmisji danych z komutacją łączy ISDN

Tabela 2. Plany rozwoju publicznych komutowanych sieci transmisji danych w Europie Zachodniej (koniec 1981 r.)

Kraj	Sieć z komutacją pakietów (PSDN)	Rok uruchomienia	Sieć z komutacją łączy (CSDN)	Rok uruchomienia
Austria	DATEX-P	1982	DATEX-L	1980
Belgia		1981		1983
Dania		1983	NPDN	1980
Finlandia		1983	NPDN	1981
Francja	TRANSPAC	1979	CADUCEE	1972
RFN	DATEX-P	1980	DATEX-L	1975
Irlandia		1983		1982
Włochy		1982		1981
Holandia	DN1	1981		
Norwegia	NORPAK	1980	NPDN	1980
Portugalia		1982		1983
Hiszpania	RETD	1971		1983
Szwecja	SWENET	1980	NPDN	1980
Szwajcaria	EDW-P	1982	EDW-A	1980
Wielka Brytania	PSS	1980	X	1982

szczególnie inteligentnymi, spowoduje, że w przyszłości będzie ona częścią sieci zintegrowanej ISDN. W tabeli 2 podano prognozę rozwoju sieci transmisji danych. Informacje te ze względu na szybki rozwój ulegają ciągłej zmianie.

PUBLICZNE SIECI TELEINFORMATYCZNE

Obecnie można rozważać trzy zasadnicze struktury publicznych sieci teleinformatycznych:

- z komutacją łączy (KŁ)
- z komutacją pakietów (KP)
- z komutacją łączy i pakietów (KŁ i KP).

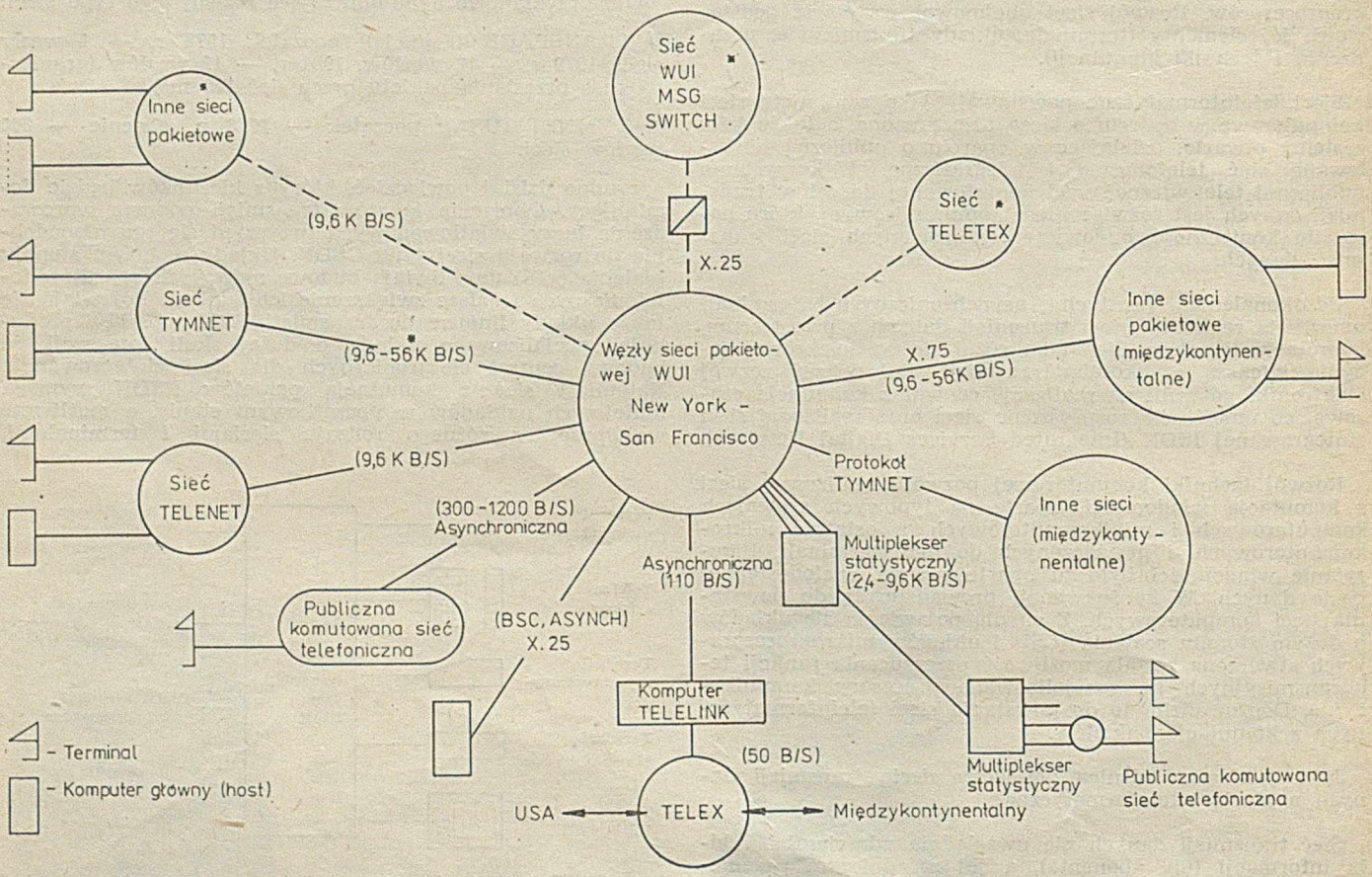
Publiczne sieci teleinformatyczne są używane w krajach wysoko rozwiniętych (USA, Japonia, RFN, Francja i inne). Najczęściej są to sieci z komutacją pakietów. Sieci z komutacją łączy występują tylko w tych krajach, które ich budowę rozpoczynały w latach sześćdziesiątych i na początku lat siedemdziesiątych. Większość sieci, których budowę rozpoczęto pod koniec lat siedemdziesiątych są z komutacją pakietów.

Na początku lat osiemdziesiątych projektowane i budowane są cyfrowe zintegrowane sieci teleinformatyczne, jak np. ISDN 1240 firmy ITT. Dla tych sieci opracowano wyposażenie, umożliwiające współpracę z centralami już istniejącymi, jak również z różnego rodzaju terminalami (według zaleceń CCITT serii X), a także systemami sygnalizacji, np. 7, a X.25 dla komunikacji pomiędzy węzłami sieci KP.

Przy rozważaniach dotyczących wyboru rodzaju sieci teleinformatycznych dla naszego kraju oraz możliwości ich realizacji, warto rozważyć nie tylko najnowsze rozwiązania, ale i te z lat siedemdziesiątych, a zwłaszcza realnie funkcjonujące. Dlatego interesujące dla nas powinny być, oprócz sieci amerykańskich typu TELENET i TYMNET, również sieci europejskie i japońskie. Z europejskich sieci na uwagę zasługują: PSS i IPSS (W. Brytania), TRANSPACK (Francja) oraz DATEX-P (RFN). Interesującymi sieciami są również DATAPAC (Kanada) i EURONET (kraje EWG) oraz VENUS (Japonia).

Podstawowymi wymaganiami użytkowników tych sieci są:

- przystosowanie do pracy z wszelkimi rodzajami terminali
- kompatybilność ze wszystkimi typami terminali
- łatwość konserwacji
- możliwość współpracy z sieciami międzynarodowymi
- obniżka kosztów dostępu do baz danych.



Rys. 2. Współpraca z innymi sieciami na przykładzie amerykańskiej sieci WUI PDSN (* — projektowane w przyszłości)

Typowa sieć KP składa się z trzech głównych elementów:

- szybkich węzłów i łącznic (koncentratorów)
- jednostek sterujących (węzłów)
- urządzeń dostępu (linii i urządzeń teletransmisyjnych) złożonych z synchronicznych linii dzierżawionych, synchronicznych modemów, synchronicznych i asynchronicznych linii specjalizowanych z multiplekserami, znormalizowanych linii dostępu telefonicznego oraz znormalizowanych linii dostępu teleksowego.

Dla terminali asynchronicznych musi istnieć w takiej sieci oprogramowanie, które zarządza procedurą składania i rozkładania pakietów. Do tego oprogramowania odnoszą się normy X.3 oraz X.25. X.25 jest normą określającą sposób, w jaki oprogramowanie steruje pracą terminali. Norma ta określa również procedurę zarządzania konwersacją między protokołem sieci. LAP i LAPB są wariantami normy X.25. LAPX jest normą, która ma umożliwić zarządzanie asynchroniczną procedurą X.25 za pośrednictwem sieci telefonicznej komutowanej.

Koncentratorów używa się jako multiplekserów umieszczonych przed centralami. Ich funkcją jest zautomatyzowanie obsługi dostępu telefonicznego oraz powiązanie różnych rodzajów terminali. Koncentratory mogą być:

- synchroniczne
- klasyczne asynchroniczne
- videotexowe asynchroniczne.

W sieciach, w których komputer nie jest w stanie stosować procedur X.25, umieszcza się przystawkę umożliwiającą adaptację komputera do wymagań tej normy.

Procedura dostępu z terminala do bazy danych w sieci KP obejmuje:

- połączenie bezpośrednie lub uzyskanie dostępu do sieci KP przez wybranie odpowiedniego numeru z komutowanej sieci telekomunikacyjnej
- dostęp do komputera w oparciu o podany adres i hasło, a następnie wybranie konkretnej bazy danych.

Przykładem bazy danych może być DBS (Database Service) uruchomiona w listopadzie 1982 r. w USA przez firmę MCI INTERNATIONAL w oparciu o sieć teleinformatyczną WUI (Western Union International). Dostęp do tej bazy danych możliwy jest także np. z polskiej sieci teleksowej. Użytkownikami tej sieci są:

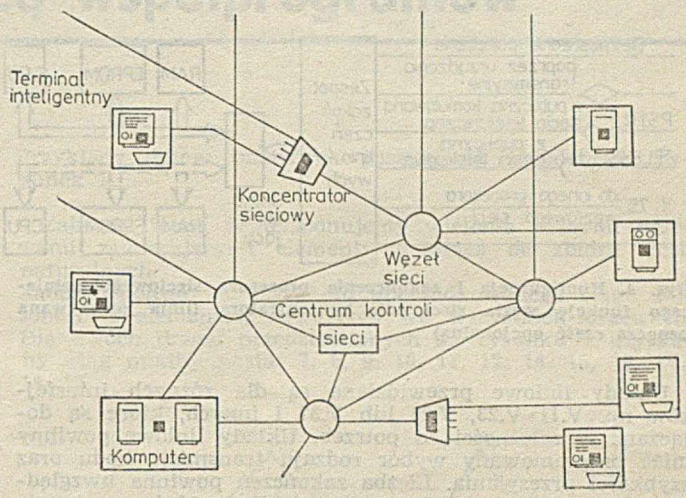
— banki
— linie lotnicze (rezerwacja miejsc)
— instytucje administracji państwowej
— instytucje i ośrodki medyczne
— biblioteki
— zakłady energetyczne
— organizacje rolnicze
— firmy transportowe
— magazyny i pośrednicy handlowi
— przemysł chemiczny
— instytucje oświatowe.

Specjalizowanym systemem, umożliwiającym wyszukiwanie danych bibliograficznych, jest również kalifornijski system DIALOG, który jest dostępny poprzez sieci TYMNET i EURONET.

Sieci KP charakteryzują się tym, iż są bardziej niezawodne i dokładne w działaniu oraz tańsze w budowie i eksploatacji od sieci KŁ. Większa niezawodność działania tych sieci wynika stąd, że pakiet może być kierowany w sieci różnymi drogami. Większa dokładność zaś stąd, że dane są sprawdzane pod kątem widzenia błędów transmisji w każdym komputerze komunikacyjnym. Wreszcie — tańsze, gdyż łącza nośne wykorzystywane są przy pełnej ich wydajności.

Sieci KP umożliwiają połączenie wielu źródeł z wieloma odbiorcami informacji. A ponieważ prawie każdy terminal sieci KP może komunikować się z każdym innym terminalem tej sieci, sieć taka nosi znamiona publicznej sieci teleinformatycznej. Budowa takiej sieci może odbywać się etapowo, a więc początkowe nakłady mogą być sprowadzone do stosunkowo niewielkich kwot.

Należy podkreślić, że w przypadku sieci KP większość operacji związanych z transmisją danych, które dawniej należały do obowiązków użytkownika, takich jak kodowanie, obsługa interfejsów, detekcja i korekta błędów oraz zapewnienie kompatybilności z różnymi komputerami i terminalami, realizowane są przez sieć i udostępniane użytkownikowi w postaci usługi. Realizacja takiej sieci w warunkach polskich wydaje się być rozwiązaniem ekonomicznym, wynikającym już z istniejących potrzeb, a także — stwarzającym niezbędną perspektywę rozwoju.



Rys. 3. Rozproszona architektura sieci na przykładzie sieci EDX-P (SIEMENS)

WYBÓR RODZAJU SIECI TELEINFORMATYCZNEJ W POLSCE

Przedstawiona poniżej propozycja obejmuje okres najbliższych kilku lat. Planowanie na okres dłuższy niż pięć lat jest niedokładne, ponieważ trudno w tej dziedzinie precyzyjnie przewidzieć dalszy rozwój technologiczny. Uwzględnia ona obecne realia i niezbędne potrzeby oraz możliwości techniczne i ekonomiczne kraju.

Wybór rozwiązania sieci teleinformatycznej wymaga uwzględnienia następujących kryteriów:

- **funkcjonalnych,** polegających na:

— dostosowaniu struktury sieci do rozmieszczenia abonentów
— elastyczności w zakresie możliwości rozbudowy sieci
— otwartej architekturze sieci (możliwość dołączania dowolnego sprzętu końcowego, tzw. sieć wirtualna)
— wzajemnym powiązaniu terminali różnych typów o różnych parametrach transmisji
— możliwości współpracy z innymi sieciami publicznymi (teleksową i telefoniczną komutowaną oraz międzynarodową siecią teleinformatyczną)
— standaryzacji i zapewnieniu uniwersalności urządzeń komunikacyjnych sieci
— jednolitym zarządzaniu całością sieci

- **sprzętowych,** polegających na:

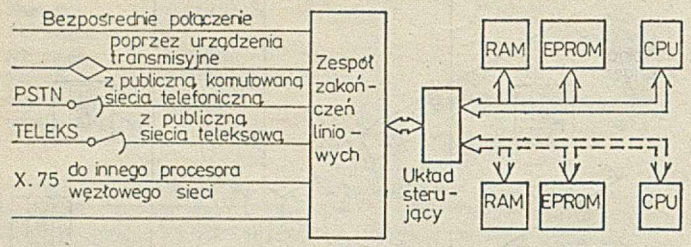
— stosowaniu sprzętu wykonanego z elementów krajowych lub krajów RWPG
— uwzględnieniu perspektyw rozwoju produkcji elementów elektronicznych

- **ekonomicznych,** polegających na:

— niewielkich nakładach w początkowym etapie rozwoju sieci
— wykorzystaniu posiadanych przez użytkowników terminali różnych typów oraz sprzętu transmisji danych (np. modemów)
— włączeniu do sieci istniejącego sprzętu komputerowego, co zapewnia lepsze jego wykorzystanie
— optymalnym wykorzystaniu łączy tranzytowych.

Powyższe kryteria spełnia sieć z komutacją pakietów. Dostępne elementy produkcji krajowej i krajów RWPG umożliwiają skonstruowanie sieci odpowiadającej aktualnemu poziomowi światowemu. Koncepcja takiej sieci zostanie poniżej przedstawiona w ogólnym zarysie. Ma ona wiele cech wspólnych z rozwiązaniami już istniejącymi.

Podstawowym elementem sieci jest węzeł (node processor) — rys. 4. Jest on wykonany w oparciu o układy mikroprocesorowe. Częścią integralną jest tu układ mikroprocesora (wraz z układami pamięci RAM i ROM), który obsługuje układy liniowe. Przewiduje się możliwość umieszczenia dwóch takich układów, z których jeden jest w stanie „gorącej" rezerwy.



Rys. 4. Konfiguracja i zakończenia procesora sieciowego pełniącego funkcję węzła sieci lub koncentratora (linia przerywana oznacza część opcjonalną)

Układy liniowe przewidziane są dla różnych interfejsów np. V.11 V.23, V.26 lub V.37 i innych, które są dołączane w zależności od potrzeb. Układy liniowe powinny mieć programowany wybór rodzaju transmisji, kodu oraz szybkości przesyłania. Liczba zakończeń powinna uwzględniać właściwości strumienia danych. Przewiduje się, że nie przekroczy ona 32.

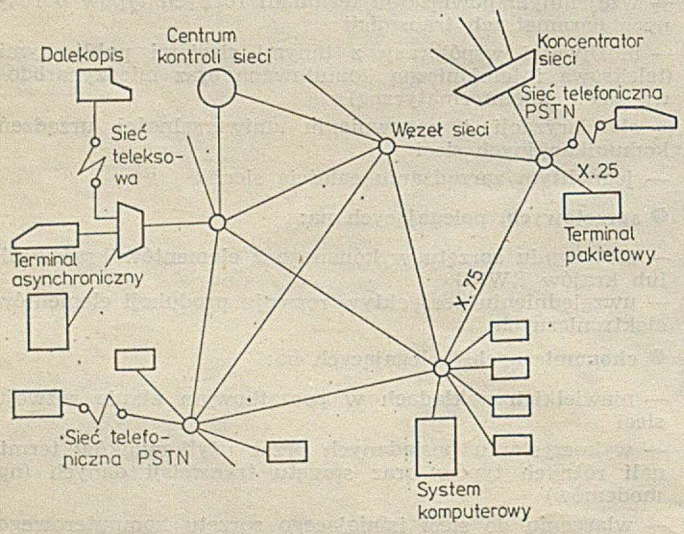
Procesor węzłowy sieci pakietowej składa się z zakończeń abonenckich lub sieciowych.

Zakończenia abonenckie mogą być dostosowane:

- do transmisji asynchronicznej start-stopowej (przyłączone bezpośrednio lub poprzez modem do sieci telefonicznej lub teleksowej)
- do transmisji synchronicznej (przyłączone bezpośrednio lub poprzez modem do sieci telefonicznej).

Zalecane są protokoły transmisji np. X.20, X.20 bis, X.21, X.25, BSC. Procesor węzłowy może mieć zakończenia abonenckie różnych typów.

Zakończenia sieciowe umożliwiają wykorzystanie łączy tranzytowych o różnych szybkościach. Jako najlepsze do tych celów należy uznać łącza szerokopasmowe 48 i 72 KB/s oraz kanał PCM 64 KB/s. Liczba zakończeń sieciowych procesora węzłowego ze względów niezawodnościowych nie powinna być mniejsza niż 3. Procesor węzłowy z jednym zakończeniem sieciowym będzie spełniał funkcje koncentratora sieciowego.



Rys. 5. Architektura pakietowej sieci teleinformatycznej i różne sposoby łączenia ze sobą jej elementów

Oprogramowanie układowe (firmware) procesora sieciowego powinno być modułowe i umożliwiać prostą modyfikację przy zmianie konfiguracji sieci. W tym zakresie można wyróżnić następujące moduły:

— procedury obsługi zakończeń różnych typów
— procedury obsługi zakończeń sieciowych
— funkcje sterujące
— opis konfiguracji sieci
— procedury komutacji pakietów (ang. routing)
— procedury diagnostyczne.

Procesory węzłowe sieci mogą być rozmieszczone (rys. 5) jako urządzenia zdalne, nadzorowane przez centrum kontroli sieci, które realizuje następujące funkcje:

— rejestracji połączeń
— taryfikacji
— statystyki
— testowania
— sygnalizacji błędów i niesprawności sieci
— sporządzania raportów o uszkodzeniach
— informacji o zmianie statusów użytkownika
— sterowania kolejkami.

* * *

Obecnie nie wystarczy już zaoferowanie dla użytkowników informatyki samych łączy. Również oferowanie publicznej sieci telefonicznej komutowanej nie może zaspokoić potrzeb, gdyż sieć ta nie jest w pełni zautomatyzowana i ma zbyt słabe parametry. Odczuwalny jest też brak systemów umożliwiających sprawne zaspokajanie potrzeb informacyjnych społeczeństwa. Dalszy rozwój teleinformatyki jest ograniczony dostępem do łączy dzierżawionych, te zaś, które już zostały wydzierżawione, są nieefektywnie wykorzystywane i dlatego, ze względów ekonomicznych, będą stopniowo ulegać likwidacji.

Aby zatem ułatwić rozwój teleinformatyki i zaspokoić realne potrzeby użytkowników na transmisję danych należałoby już dziś przystąpić do budowy krajowej sieci teleinformatycznej. Jaka to powinna być sieć? Aby precyzyjnie odpowiedzieć na to pytanie należałoby przeprowadzić odpowiednie badania i zebrać niezbędne informacje. Czy to możliwe? Prace badawcze prowadzone były zbyt długo nad „Infostradą" i brak jest dotąd wniosków z tych badań. Nie ma też dziś możliwości zebrania informacji, w oparciu o które można by było przystąpić do projektowania publicznej sieci teleinformatycznej.

Należy zatem rozważyć możliwość oparcia się na doświadczeniach innych krajów. Należy szukać analogii między naszą sytuacją a sytuacją w tych krajach, jak również dokonać takiego wyboru typu sieci, by nie powtarzać błędów i dokonać przeskoku do nowoczesnych rozwiązań, uwzględniając spodziewany postęp techniczny, a także nasze obecne przygotowanie i stan posiadania.

Trzeba rozważyć możliwość skomutowania w pierwszym etapie łączy dzierżawionych oraz wykorzystać prace uruchomione pod kątem widzenia komutacji łączy. Sytuacja ta jakby więc wymusza realizację sieci z komutacją łączy. Biorąc jednak pod uwagę — występujący w końcowych latach siedemdziesiątych i w początkach lat osiemdziesiątych — trend budowy sieci z komutacją pakietów, powinno się zatem preferować ten rodzaj sieci. Takie podejście wskazuje, że nie powinniśmy rezygnować z żadnej możliwości, a więc podobnie jak w latach siedemdziesiątych wybraliśmy dwa typy licencyjnych central telefonicznych, tak i teraz należałoby chyba wybrać realizację dwóch typów sieci teleinformatycznych, a mianowicie z komutacją łączy (KŁ) i komutacją pakietów (KP).

Powinno się zatem przystąpić do budowy wzajemnie się uzupełniających sieci z KŁ i KP. Przyszłość wykaże, która z tych koncepcji sieci będzie dla naszych polskich warunków lepsza. Należy również przewidzieć możliwość współpracy naszej sieci z sieciami innych krajów oraz — że powinna być ona siecią publiczną. W oparciu o taką sieć powinny być organizowane sieci resortowe i wydzielonych grup użytkowników.

WIESŁAWA BARTOL
Instytut Informatyki
Uniwersytet Warszawski

Programowanie za pomocą współprogramów

Współprogramy (ang. coroutines) są bardzo silnym narzędziem programowania, ułatwiającym pisanie programów o strukturze modularnej w sekwencyjnym języku programowania. Nazwa „coroutine" została po raz pierwszy użyta przez Conway'a [2] do określania takich części składowych programu, które mogą współpracować ze sobą w trakcie wykonywania programu. Współpraca ta polega na wzajemnym przekazywaniu sterowania między współprogramami. Conway porównał współprogramy do „wzajemnych podprogramów" (ang. mutual subroutines), czyli takich podprogramów, które same sterują wykonywaniem swoich akcji bez żadnego programu głównego.

Niniejszy artykuł jest próbą sprecyzowania jaki moduł programu będzie nazywany współprogramem oraz określenia jego własności, prezentuje też przykłady algorytmów zapisanych przy użyciu współprogramów. Użyta zostanie notacja zaczerpnięta z języka LOGLAN [5], który posiada środki pozwalające na programowanie za pomocą współprogramów. Współprogram możemy porównać z procedurą lub klasą. Podobnie jak procedura i klasa, posiada on lokalną strukturę danych oraz ciąg instrukcji. Różnica uwidocznia się dopiero w trakcie wykonywania instrukcji umieszczonych w treści współprogramu.

Wykonanie instrukcji może zostać w pewnym miejscu przerwane przez zgłoszenie chęci skomunikowania się z innym współprogramem. W takiej sytuacji sterowanie programu opuszcza aktywny dotąd współprogram i przenosi się do innego wpsółprogramu, który z kolei staje się aktywny. Akcje nowo aktywowanego współprogramu zostają wznowione od miejsca, w którym uprzednio zostały zatrzymane. Sytuacja taka nie jest możliwa w przypadku procedur, aczkolwiek można przerwać wykonywanie akcji procedury np. przez wywołanie innej procedury lub wykonanie instrukcji **goto**, natomiast nie ma możliwości zaprogramowania wznowienia przerwanej procedury z innego miejsca programu.

Współprogram może mieć rozbudowaną strukturę tzn. mogą być w nim zadeklarowane procedury, inne współprogramy itp. Jego treść może mieć strukturę blokową. Przerwanie wykonywania akcji współprogramu może nastąpić w dowolnym miejscu w jego treści np. w podbloku, w wywołanej procedurze (nawet rekurencyjnej) lub funkcji. Takie przerwanie zawsze dotyczy całego współprogramu.

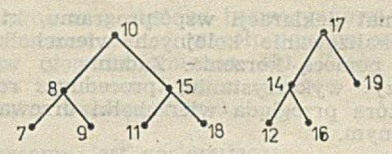
PRZYKŁAD 1

Niniejszy przykład zastosowania współprogramów został zaczerpnięty z [8]. Załóżmy, że pewien zbiór danych, w którym jest określony liniowy porządek, został przedstawiony za pomocą drzewa binarnych poszukiwań. Drzewo binarnych poszukiwań przedstawiające zbiór danych — to drzewo binarne spełniające następujące warunki:

- elementy zbioru danych są umieszczone w wierzchołkach drzewa
- każdy wierzchołek u należący do lewego poddrzewa wierzchołka v spełnia relację $u < v$
- każdy wierzchołek u należący do prawego poddrzewa wierzchołka v spełnia relację $u > v$
- dla każdego elementu zbioru danych istnieje dokładnie jeden wierzchołek drzewa zawierający ten element.

Przykłady drzew binarnych poszukiwań przedstawia rysunek 1.

Załóżmy teraz, że dysponujemy wieloma różnymi drzewami zawierającymi elementy należące do zbioru liczb naturalnych.
Zadanie polega na uzyskaniu uporządkowanego ciągu złożonego z elementów zawartych we wszystkich drzewach. Dla dwóch drzew przedstawionych na rysunku 1 uzyskany ciąg miałby posta: 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19.



Rys. 1. Drzewa binarnych poszukiwań przedstawiające odpowiednio zbiory: {7, 8, 9, 10, 11, 15, 18} oraz {12, 14, 16, 17, 19}

ALGORYTM

Elementy dołączone do ciągu w kolejnym kroku algorytmu są wybierane jako minimalne spośród tych wierzchołków drzew, które nie były dotąd brane pod uwagę. W pierwszym kroku zlokalizowane zostają elementy minimalne we wszystkich drzewach (w każdym z drzew jest to najbardziej na lewo położony liść) oraz wybrany jest najmniejszy spośród nich — stanowi on pierwszy element ciągu.

W kolejnym kroku w drzewie, z którego pochodził ostatnio dołączony element, lokalizujemy następny co do wielkości. Wybieramy najmniejszy spośród tych zlokalizowanych we wszystkich drzewach i dołączamy go do ciągu. Powtarzamy tę procedurę, aż do wyczerpania wszystkich wierzchołków drzew. Zauważmy, że znajdowanie kolejnych co do wielkości elementów w drzewie polega na przeglądaniu drzewa w porządku wewnętrznym (ang. inorder). Dobrze byłoby obchodzić każde drzewo w tym porządku oraz zatrzymać się po zlokalizowaniu kolejnego wierzchołka. Zatrzymanie pozwoliłoby wybrać element najmniejszy ze wszystkich aktualnie zlokalizowanych w drzewach i dołączyć go do ciągu.

W jaki sposób można zastosować współprogramy do zaprogramowania tego algorytmu? Rozwiązanie polega na użyciu współprogramu związanego z każdym drzewem. Jego zadaniem jest obchodzenie drzewa i zatrzymanie się po znalezieniu kolejnego elementu. Sterowanie jest wówczas przekazywane do współprogramu wybierającego element, który będzie dołączony do ciągu.

Przed napisaniem programu realizującego to zadanie trzeba wiedzieć, jak reprezentować w programie współprogram oraz jakie są instrukcje pozwalające na komunikowanie się współprogramów między sobą.

POSTAĆ DEKLARACJI WSPÓŁPROGRAMU

Syntaktyczna postać deklaracji współprogramu w różnych językach programowania może przybierać różne formy.

Opiszemy postać zaczerpniętą z języka LOGLAN. Schemat deklaracji jest następujący:

```
unit nazwa współprogramu:
                    coroutine
                    (parametry formalne);
                    deklaracje lokalnych wielkości;
                    begin
                    instrukcje współprogramu
                    end
```

Łatwo zauważyć, że deklaracja współprogramu różni się od deklaracji klasy tylko zastąpieniem słowa **class** przez słowo **coroutine.** Podobnie jak w przypadku klas, deklaracja stanowi wzorzec, według którego tworzone są dynamiczne jednostki (obiekty). Jednostki takie będziemy nazywać obiektami przedstawiającymi współprogramy lub współprogramami. Można oczywiście utworzyć wiele współprogramów na wzór tej samej deklaracji. Będziemy mówić, że są one tego samego typu.

Rozważmy np. deklarację:

```
unit współprogramu: coroutine korzeń: wierzchołek);
                    unit obchodź: procedure
                                  w:wierzchołek);
                                  ...
                                  end obchodź;
                                  begin
                                  return;
                                  call obchodź (korzeń)
                    end współprogramu
```

Jest to schemat deklaracji współprogramu, którego zadaniem jest lokalizowanie kolejnych wierzchołków drzewa zadanego za pomocą korzenia. Zadanie to współprogram realizuje przy wykorzystaniu procedury rekurencyjnej „obchodź", która przegląda wierzchołki drzewa w porządku wewnętrznym.

Pierwszą instrukcją treści współprogramu jest instrukcja **return.** Jest to instrukcja odpowiedzialna za utworzenie współprogramu, tzn. od momentu jej wykonania współprogram istnieje jako samodzielny obiekt. Szerzej ją omówimy przy opisie generowania współprogramu.

Następną instrukcją jest wywołanie procedury przeglądającej drzewo. Ponieważ żądamy, aby akcje były zatrzymywane po odwiedzeniu każdego wierzchołka, zatem odpowiednia instrukcja zatrzymująca będzie umieszczona w treści tej procedury.

GENEROWANIE WSPÓŁPROGRAMU

Generowanie współprogramu następuje jako efekt obliczenia wartości wyrażenia generującego postaci:
new nazwa współprogramu (parametry aktualne)

Wartością takiego wyrażenia jest obiekt przedstawiający współprogram utworzony według odpowiedniej deklaracji. W momencie obliczania wartości wyrażenia generującego tworzy się odpowiedni obiekt, do którego przekazane zostają parametry aktualne oraz rozpoczyna się wykonywanie akcji umieszczonych w treści współprogramu.

Mówimy, że współprogram został utworzony w momencie wykonania instrukcji **return.** Powoduje ona przerwanie wykonywania akcji współprogramu i przekazanie sterowania do miejsca następującego za generatorem.

Opiszemy teraz instrukcje pozwalające na komunikowanie się współprogramów między sobą.

KOMUNIKACJA WSPÓŁPROGRAMÓW

Instrukcje pozwalające na przekazywanie sterowania między współprogramami są następujące:

- **attach** (X), gdzie X jest nazwą współprogramu
- **detach.**

Instrukcja **attach** (X) wykonana w treści współprogramu **W** powoduje, że sterowanie opuszcza współprogram **W** i przenosi się do współprogramu **X,** którego akcje są wznowione dokładnie od miejsca, gdzie uprzednio zostały wstrzymane.

Wykonanie instrukcji **detach** w treści współprogramu **W** powoduje wstrzymanie akcji obiektu **W** i przeniesienie sterowania do współprogramu, który uprzednio spowodował uaktywnienie **W** przez wykonanie **attach (W).**

Mówimy, że współprogram jest aktywny, gdy są wykonywane jego instrukcje. W momencie gdy sterowanie zostaje mu odebrane, przechodzi on w stan oczekiwania na ponowne uaktywnienie. Znaczenie instrukcji **attach** oraz **detach** ilustruje rysunek 2. Rysunek ten przedstawia pewien stan wykonania programu, w którym utworzono dwa współprogramy nazwane odpowiednio **W** oraz **X.** Współprogram **W** jest aktywny, ponieważ sterowanie programu (SP) wskazuje pewną jego instrukcję (jest nią instrukcja **attach** (X)). Współprogram **X** ma wstrzymane akcje. Jego sterowanie lokalne (SL) wskazuje, od którego miejsca współprogram ten będzie uaktywniony.

Załóżmy teraz, że następuje wykonanie instrukcji **attach** (X) w obiekcie **W.** W rezultacie współprogram **X** stanie się aktywny. Rysunek 3 ilustruje tę sytuację. Natomiast wykonanie instrukcji **detach** w obiekcie **X** prowadzi do konfiguracji pokazanej na rysunku 4.

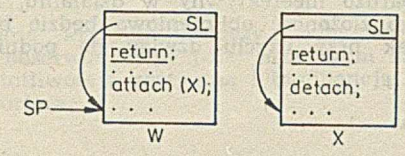
Blok całego programu jest również traktowany jako współprogram (o nazwie **main).** Przekazanie do niego sterowania można uzyskać dzięki wykonaniu instrukcji **attach (main).**

PROGRAM DLA PRZYKŁADU 1

Napiszemy teraz schemat całego programu „łączenia" drzew, opisanego wcześniej. Dla uproszczenia programu przyjmiemy założenie, że wszystkie liczby umieszczone w wierzchołkach drzew są mniejsze od pewnej liczby **M.**
Schemat programu jest następujący:

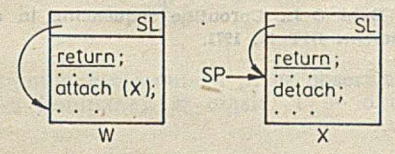
```
program
    unit wierzchołek: class;
                      var lewy, prawy: wierzchołek,
                          wart: integer
                      end wierzchołek;
    var drzewa: array of wierzchołek; (* drzewa
                                  binarnych poszukiwań *)
    unit współprogram: coroutine (korzeń: wierzchołek);
                       var aktmin: integer;
                       unit obchodź: procedure
                                     (w:wierzchołek);
                                     begin
                       wart: integer
                       end wierzchołek;
                       if w = none then call
                            obchodź (w. lewy);
                            aktmin: = w. wart;
                       if w ≠ none then call
                       (*zmienna aktmin przekazuje
                       zlokalizowaną wartość *)
                            detach;
                            call obchodź
                                 (w. prawy)
                       fi
                       end odchodź;
            begin
            return;
            call obchodź (korzeń); aktmin: = M
            end współprogram;
var wsp: array of współprogram, i, n, M, min: integer;
begin
   read (n); (* wczytanie liczby drzew *)
   read (M);
array drzewa dim [1..n];
array wsp dim [1..n];
(* utworzenie dwu tablic długości n *)
for i: = 1 to n do drzewa [i]: = ... od;
(* utworzenie n drzew binarnych poszukiwań
  o wierzchołkach przechowujących liczby naturalne *)
for i: = 1 to n do wsp [i]: = new współprogram
  (drzewa [i]); od;
(* utworzenie n współprogramów *)
for i: = to n do attach (wsp [i]); od;
(* każdy współprogram zlokalizował najbardziej
  lewy liść w drzewie *)
min: = 0;
while min < M do
      min: = minimum (drzewa [1] aktmin,...,
      drzewa [n]. aktmin);
        i: = numer drzewa zawierającego minimalną
        wartość wyznaczoną w poprzedniej instrukcji;
      if min < M then write (min); attach (wsp. [i]) fi;
      od
end program
```

Tworzony ciąg liczb jest wypisywany instrukcją **write (min)**. W programie powstaje **n** współprogramów, których nazwami są odpowiednie elementy tablicy **wsp**. Sterowanie programu jest przekazane do każdego z nich dzięki instrukcji **attach** (wsp. [i]). Akcje każdego współprogramu są zatrzymane dzięki **detach** w momencie zlokalizowania najmniejszej wartości w każdym drzewie.



Rys. 2. Przykład współprogramów

W pętli while min < M **do**... blok całego programu komunikuje się ze współprogramami w celu wybierania i wypisywania kolejnych elementów ciągu. Po wpisaniu kolejnej liczby sterowanie zostaje przekazane do współprogramu, który zlokalizował tę liczbę w swoim drzewie. Współprogram ten zlokalizuje następną co do wielkości wartość, a następnie dzięki instrukcji **detach** odda sterowanie z powrotem do bloku **program**.



Rys. 3 Przekazanie sterowania do współprogramu X

PRZYKŁAD 2

W drugim przykładzie współdziałać będą dwa współprogramy tego samego typu, czyli według tej samej deklaracji. Przypuśćmy, że w danym tekście każde wystąpienie pary liter **aa** chcemy zastąpić literą **b**, a następnie w tak przerobionym tekście zastąpić każde wystąpienie pary **bb** przez literę **c**. Na przykład słowo **caabd** zostanie zamienione na **ccd**, a słowo **aaaabb** na **cc**.

W programie realizującym to zadanie użyte będą trzy współprogramy. Jeden odpowiedzialny za czytanie tekstu z wejścia, drugi za zamianę **aa** na **b**, a trzeci za zamianę **bb** na **c**. Wszystkie trzy komunikują się za pomocą pewnej zmiennej globalnej. Pierwszy czyta kolejne znaki i przez tę zmienną przekazuje je do drugiego współprogramu. Ten z kolei przekazuje je do trzeciego współprogramu dokonując zamiany **aa** na **b**. Trzeci współprogram przekazuje kolejne znaki do bloku programu (dokonując zamiany **bb** na **c**), który z kolei je drukuje.

Można zauważyć, że treści współprogramów drugiego i trzeciego różnią się tylko literami, które zamieniają, oraz współprogramami, z którymi współpracują (drugi odbiera znaki od pierwszego, a trzeci od drugiego). Można zatem napisać wspólną treść dla tych współprogramów, a odpowiednie litery i współprogramy uczynić parametrami. Napiszemy deklarację współprogramu, który będzie dokonywać odpowiedniej zamiany liter. Jego nagłówek jest następujący:

```
unit zamiana: coroutine (11, 12: character,
                         wejście: coroutine);
```

Współprogram ten ma zadanie dokonać odpowiedniej zamiany liter w tekście dostarczonym mu przez współprogram odpowiadający parametrowi **wejście**. **Wejście** dostarcza kolejne litery tekstu, w którym każde wystąpienie pary liter **l1 l2** ma być zastąpione literą **l2**. Współprogramy będą się komunikować przez zmienną globalną **ch**.

```
unit zamiana: coroutine l1, l2: character,
                        wejście: coroutine);
              var pom: character;
              begin
              return;
              do
              attach (wejście); (* komunikacja z wejściem
                                   w celu otrzymania kolej-
                                   nego znaku tekstu, będą-
                                   cego wartością zmien-
                                   nej ch *)
              if ch = l1 then (* wystąpiła litera l1 —
                                 należy sprawdzić następ-
                                 ną literę *)
                 attach (wejście); (* wejście dostarcza
                                      następną literę *)
                 if ch = l1 then (* wystąpiła druga lite-
                                    ra l1 *)
                                 ch: = l2
                            else (* druga litera jest różna
                                    od l1, należy ją zapamiętać
                                    na zmiennej pom *)
                                 pom: = ch; ch: = l1;
                                 detach,
                                 ch: = pom;
              fi; fi;
              detach (* oddanie sterowania *)
              od
              end zamiana;
```

Treść współprogramu stanowi bezwarunkowa pętla: **do...od**, w której odbierane są od współprogramu **wejście** kolejne znaki tekstu (poprzez zmienną **ch**). Znaki te podlegają ewentualnym zamianom i przez zmienną **ch** przekazywane są na zewnątrz współprogramu.

Treść programu jest następująca:

```
program wymiana — liter
   unit zamiana: coroutine (l1, l2: character,
                            wejście: coroutine);
                 ...
                 end zamiana;
   unit czytatekst: coroutine: (* współprogram czytający
                                  tekst *)
                 begin
                 return;
                 do
                 read (ch); detach
                 od
                 end czytatekst;
var ch, ostatniznak: character,
        nastznak: czytatekst,
        AB, BC: zamiana;
begin
   read (ostatniznak); (* przeczytanie znaku
                           kończącego tekst *)
   nastznak: = new czytatekst;
   AB: = new zamiana ('a', 'b', nastznak);
   BC: = new zamiana ('b', 'c', AB);
   do
attach (BC); write (ch); if ch = ostatniznak
                                    then exit fi
   od
end wymiana — liter
```

W programie głównym zostają utworzone trzy współprogramy o nazwach odpowiednio: nastznak, AB, BC. Sterowanie najpierw przechodzi do współprogramu BC, ten z kolei uaktywnia współprogram AB, który dostaje litery od współprogramu **nastznak**.

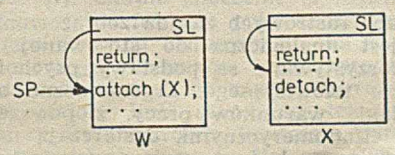
Współprogram AB powstaje w rezultacie wykonania instrukcji:

```
AB: = new zamiana ('a', 'b', nastznak);
```

gdzie **nastznak** jest nazwą współprogramu czytającego tekst wejściowy i dostarczającego kolejne jego znaki jako wartości zmiennej globalnej **ch**.

Współprogram BC tworzymy:

```
BC: = new zamiana ('b', 'c', AB);
```



Rys. 4. Stan programu po wykonaniu instrukcji detach w X

Można zauważyć, że współprogram BC będzie korzystał z tekstu przekazywanego mu przez współprogram AB.

Wykonanie instrukcji **detach** w obiekcie BC spowoduje oddanie sterowania do programu głównego, gdzie znak będący wartością **ch** jest drukowany.

Pojawienie się znaku kończącego tekst powoduje wyjście z bezwarunkowej pętli w programie głównym dzięki instrukcji **exit** i zakończenie programu.

* * *

Wykonywanie programu wykorzystującego współprogramy nazywa się czasem wykonywaniem „quasi-równoległym". Jest to uzasadnione własnością współprogramów, umożliwiającą symulowanie równoległych procesów w ramach jednego programu sekwencyjnego. Niektóre języki programowania (SOL, SIMULA-67, LOGLAN) wykorzystują tę własność współprogramów do tworzenia specjalnych mechanizmów umożliwiających pisanie programów symulacyjnych.

Współprogram może czasem zastąpić procedurę. W przypadku wielokrotnego wywoływania w programie tej samej procedury można jej deklarację zastąpić deklaracją odpowiedniego współprogramu. Treść współprogramu stanowi wówczas bezwarunkowa pętla, a instrukcja **detach** (przed końcem pętli) powoduje przekazanie sterowania w miejsce zastępujące wywołanie procedury. Wywołanie procedury jest zastąpione instrukcją **attach** (W), gdzie **W** jest współprogramem (ewentualne przekazanie parametrów aktualnych można zastąpić przez zmianę wartości atrybutów współprogramu). Procedura wymaga każdorazowo rezerwowania nowego pola w pamięci maszyny, podczas gdy pole dla współprogramu rezerwowane jest tylko raz — przy generacji.

Współprogramy są bardzo silnym narzędziem programowania, ale nieumiejętne lub nadmierne ich stosowanie może spowodować, że program — choć elegancko napisany — okaże się bardzo nieefektywny w działaniu, inaczej mówiąc — jego złożoność obliczeniowa będzie bardzo duża. Podobnie jak przy użyciu dźwigu do podniesienia kamyczka...

LITERATURA

[1] Britwistle G., Dahl O-J., Myhrhaug B., Nygaard K.: SIMULA Begin. Studentliteratur Auerbach Publishers, Philadelphia, Lund, 1974

[2] Conway M. E. et al: Design of a Separable Transition-Diagram Compiler. Comm. ACM6, July 1963

[3] Dahl O. J., Myhrhaug B., Nygaard K.: SIMULA-67 Common Base Language. Norwegian Computing Center, 1970

[4] Marlin Ch.: Coroutines. Springer-Verlag, Lectures Notes in Computer Science, 1980

[5] Muldner T i in.: LOGLAN-raport. Maszynopis

[6] Oktaba H., Ratajczak-Bartol W.: Realizacja programu w SIMULI-67 Sprawozdania Instytutu Informatyki UW nr 58, 1977

[7] Oktaba H., Ratajczak-Bartol W.: SIMULA 67. WNT, Warszawa, 1980

[8] Wang A., Dahl O-J.: Coroutine sequencing in a block structured environment. BIT 11, 1971.

Efektywność systemów graficznych, usprawniających dialog człowieka z komputerem, zależy w dużej mierze od stopnia, w jakim uwzględniono przy ich konstrukcji zdolności percepcyjne użytkownika. Z tego względu w piątym (ostatnim) artykule prezentowanej serii, autorstwa Jacka Owczarczyka, omawiane są szerzej zagadnienia percepcji wzrokowej.

W dziale TERMINOLOGIA (s. 37) zamieszczamy ponadto słowniczek terminów angielskich związanych z systemami graficznymi; zawiera on pojęcia, które w większości omówione zostały w artykułach prezentowanej serii. (Red).

JACEK OWCZARCZYK

Warszawa

Systemy graficzne: percepcja wzrokowa

Ze wszystkich zmysłów człowieka, rozpatrywanych w kategoriach kanałów przesyłania informacji, wzrok wykazuje największą efektywność — niemal 80% informacji odbieranej z otaczającego nas świata dociera przez system wzrokowy. Zrozumiałe jest więc wykorzystywanie najlepszego z naszych zmysłów w dialogu z komputerem i zrozumiałe — olbrzymie zainteresowanie systemami graficznymi.

Wprawdzie koszt systemów graficznych uległ znacznemu obniżeniu i są one coraz bardziej rozpowszechnione, niemniej istnieje ciągle pewna granica dokładności obrazowania przez nie informacji graficznej. Przekroczenie tej granicy jest możliwe obecnie za cenę przewyższającą finansowe możliwości przeciętnych użytkowników. Badania naukowe natomiast coraz bardziej przekonują, że właściwe wykorzystywanie psychofizjologicznych właściwości naszego systemu wzrokowego pozwala zwiększyć skuteczność odbioru informacji graficznej stosunkowo niedużym kosztem. Dlatego też pewien podstawowy zasób wiadomości o systemie wzrokowym jest niezbędny dla projektantów, konstruktorów i użytkowników systemów graficznych.

W systemach rastrowych zasada zobrazowania informacji graficznej jest analogiczna do stosowanej w telewizji, a więc wykorzystywane są podstawy psychofizjologii widzenia obrazu telewizyjnego. Ergonomiczne aspekty telewizji i badania warunków pracy, z powszechnymi dziś terminalami alfanumerycznymi, dostarczają najwięcej rozwiązań, które mogą być stosowane również w systemach graficznych.

BEZWŁADNOŚĆ WZROKU

Jedną z właściwości systemu wzrokowego, która wykorzystywana jest w odbiorze obrazu telewizyjnego, jest bezwładność wzroku, tzn. reagowanie z pewnym opóźnieniem na pojawienie się i zniknięcie światła. Cykliczne powtarzanie obrazu z wystarczająco dużą częstotliwością (wyższą od częstotliwości krytycznej) powoduje, że wrażenie migotania obrazu zanika.

Częstotliwość krytyczna zależy m.in. od luminancji i wielkości obserwowanego pola, przy czym zależność pomiędzy częstotliwością krytyczną i luminancją ujmuje prawo Ferry-Portera:

$$f_k = a \log L + b$$

gdzie: f_k — krytyczna częstotliwość migotania, L — luminancja, a i b — stałe zależne od warunków obserwacji (tab. 1).

Zmiany luminancji z częstotliwością wyższą od krytycznej częstotliwości migotania nie są dostrzegane, a luminancję ocenianą przez obserwatora określa prawo Talbota:

$$L_0 = \frac{1}{T} \int_0^T L(t)\,dt$$

gdzie L_O — luminancja oceniana przez obserwatora, L(t) — funkcja zmian luminancji w czasie, T — okres zmian.

PERCEPCJA KONTRASTU

Oko jest w stanie rozróżnić dwa obiekty, jeżeli kontrast ich luminancji ma dostateczną wartość liczbową. Próg kontrastu luminancji, poniżej którego obiekty te są nierozróżnialne, zależy od wielu czynników, takich jak: rozmiary obserwowanych obiektów, czułości oka, luminancji tła i konfiguracji obiektów, a także od czasu obserwacji (adaptacji oka). Udział tego ostatniego czynnika wynika z przystosowywania się systemu wzrokowego do warunków obserwacji. W szczególności — próg kontrastu luminancji ma najniższą wartość, jeżeli przed obserwacją wzrok obserwatora nie odbierał fal świetlnych (adaptacja w ciemni).

W szerokim zakresie luminancji I, dostrzegalną różnicę luminancji tła i obserwowanego obiektu ΔI opisuje prawo Webera-Fechnera, które mówi, że dostrzegalna różnica luminancji ΔI jest proporcjonalna do I, tzn. ΔI musi być tym większe, im większa jest luminancja tła I.

Ten logarytmiczny charakter systemu wzrokowego (wrażenie kontrastowości zależy od różnicy logarytmów luminancji) został uwzględniony w jednej z metod zwiększania kontrastu obrazów cyfrowych. W metodach tych dąży się zwykle do modyfikacji histogramu częstości występowania poszczególnych poziomów szarości (stopni kontrastu) w celu uzyskania rozkładu zbliżonego do rozkładu jednostajnego. Uzasadnieniem tego postępowania jest maksymalizowanie entropii, która uważana jest za miarę informacji przenoszonej przez obraz. Frei zwrócił uwagę [2], że w sytuacjach, gdy obraz obserwowany jest przez człowieka, warunki odbioru są optymalne (z uwagi na prawo Webera-Fechnera) dla innej modyfikacji histogramu — mianowicie takiej, która powoduje, że entropia odebranego (a nie obserwowanego) obrazu jest największa.

Szacuje się, że użytkownik systemu graficznego jest w stanie rozróżnić na ekranie monitora ok. 50 stopni kontrastu (ich liczba w bardzo dobrym obrazie telewizyjnym wynosi ok. 100 [7]. Istotne znaczenie w wyświetlaniu informacji graficznej ma fakt, że próg kontrastu luminancji dla jasnych rysunków na ciemnym tle ma wartość niższą niż próg dla ciemnych rysunków na jasnym tle. Przykładowo — czytelność elementów mających rozmiary odpowiadające grubościom linii standardu telewizyjnego 625 linii uzyskuje się przy wzroście kontrastu odpowiednio 32 razy (jasne elementy na ciemnym tle) oraz 100 razy (ciemne elementy na jasnym tle) [7].

Tabela 1. Wartości stałych a i b [7]

Migotanie	a	b
Niezauważalne	9,61	65,15
Dopuszczalne	8,67	58,70
Przykre	7,50	51,00

PERCEPCJA KOLORÓW

Obraz kolorowy tworzony jest na ekranie monitora przez sumowanie w odpowiedniej proporcji trzech podstawowych kolorów (czerwony, zielony, niebieski). Dla każdego z podstawowych kolorów istnieje w kineskopie kolorowym oddzielne działo elektronowe powodujące pobudzenie luminoforu o odpowiedniej barwie poświaty.

Zasadę postrzegania kolorów opisuje trójbodźcowa teoria Younga, według której w siatkówce oka istnieją trzy różne odmiany czopków (fotoreceptorów). Reagują one na światło czerwone, zielone lub niebieskie, przy czym odpowiednie długości fal świetlnych wynoszą: dla światła czerwonego — 580—700 nm, zielonego — 490—580 nm oraz niebieskiego — 400—490 nm.

Reakcję oka na światło o różnych barwach opisuje spektralna charakterystyka względnej czułości oka. Zarówno czułość oka, jak i ostrość widzenia zależy od długości fali świetlnej. Dla długości fal odpowiadającym luminoforom — stosowanym w telewizji kolorowej — względną wartość czułości oka i ostrości widzenia przedstawiono w tabeli 2 [7].

Światło o barwie zielonej przyjmowane jest przez wzrok z najmniejszym wysiłkiem. Właściwość ta wykorzystywana jest praktycznie w konstrukcji terminali alfanumerycznych — bądź przez stosowanie luminoforów o zielonej barwie świecenia, bądź przez pokrywanie typowych kineskopów specjalną powłoką zmieniającą barwę obrazu na zieloną.

Użytkownik jest w stanie rozróżnić na ekranie monitora ok. 160 różnych kolorów [1], a więc znacznie więcej niż poziomów szarości (stopni kontrastu). Wskazuje to na większą efektywność monitorów kolorowych w systemach graficznych. Pewne ilościowe oszacowanie różnicy pomiędzy kolorowym i monochromatycznym przetwarzaniem obrazów wynika z porównania sprawności systemów telewizji kolorowej i czarno-białej. Wykorzystując aparat teorii informacji, można wykazać, że sprawność telewizji kolorowej jest trzykrotnie większa [7].

Warto podkreślić, że w zobrazowaniach dwukolorowych następuje pogorszenie rozdzielczości wzroku. Jeżeli przyjąć, że wynosi ona 100% dla zobrazowania czarno-białego, to względna rozdzielczość dla zobrazowania czerwono-zielonego wynosi 40%, dla czerwono-niebieskiego — 23%, a dla zielono-niebieskiego jedynie 19% [7].

Kontrast luminancji jest lepiej postrzegany niż barwa czy kształt. W miarę zmniejszania luminancji obiektu o określonej barwie i kształcie, obserwator przestaje rozróżniać kolejno:

— barwę; przedmiot staje się szary (o określonych kształtach), przekroczony zostaje próg kontrastu barwy

— kształt; przedmiot wydaje się plamą o nieokreślonych konturach, przekroczony zostaje próg rozpoznawania kształtów

— kontrast luminancji; przy przekroczeniu progu kontrastu luminancji plama i tło są nierozróżnialne.

Tabela 2. Względna wartość czułości oka i ostrości widzenia

Kolor	Względna czułość oka (100% dla 555 nm)	Względna ostrość widzenia (100% dla 565 nm)
Czerwony (610 nm)	47%	60%
Zielony (535 nm)	92%	70%
Niebieski (470 nm)	17%	12%

STEREOSKOPIA W SYSTEMACH GRAFICZNYCH

Wysiłki konstruktorów systemów graficznych skoncentrowane były (i są) na stworzeniu możliwości generowania na ekranie monitora obrazów jak najbardziej realistycznych, przy czym realizm oznacza tu maksymalną skuteczność odbioru informacji graficznej tworzonej za pomocą komputera. Krokiem w tym kierunku było wprowadzenie rastrowych systemów graficznych, w których istnieją techniczne możliwości tworzenia obrazów zbliżonych do rzeczywistych. Kolejnym etapem jest wprowadzanie metod dających użytkownikom systemów graficznych wrażenie trójwymiarowości oglądanych na ekranie przedmiotów i ich wzajemnego przestrzennego rozmieszczenia.

Należy podkreślić, że uzyskiwanie wrażenia trójwymiarowości jakkolwiek stosowane jest często w czysto spektakularnych celach, ma istotne znaczenie w wielu zastosowaniach. W medycynie — na przykład — stwarza to możliwości wykorzystania systemów graficznych w leczeniu pewnych chorób oka [4] oraz pozwala kardiologom obserwować wewnętrzne struktury serca [3]. Symulator lotów, nie dający pilotowi możliwości oceny przestrzennego rozmieszczenia przedmiotów, ma ograniczoną przydatność, zwłaszcza podczas treningu lądowania w trudnych warunkach (np. na pokładzie lotniskowca).

Efekt trójwymiarowości wyświetlanych na ekranie przedmiotów można uzyskać kinetycznie. Zbliżenia i obroty, widziane często na ekranach grafoskopów, pozwalają ocenić przestrzenne relacje poszczególnych elementów sceny. W rastrowych systemach graficznych istnieją ponadto możliwości zastosowania różnych metod stereoskopowych, które pozwalają uzyskać pełny efekt widzenia przestrzennego.

Widzenie przestrzenne, dające możliwość oceny względnej odległości pomiędzy poszczególnymi przedmiotami, zapewnia nam obserwacja dwuoczna. Każde oko dokonuje niezależnie akomodacji (ustawienie ostrości) i odpowiedniego obrotu. Dlatego też na siatkówce każdego oka tworzy się inny obraz obserwowanej sceny. Różnica pomiędzy tymi obrazami stanowi źródło informacji o przestrzennych relacjach.

W systemach graficznych wrażenie widzenia przestrzennego uzyskuje się generując dwa obrazy, z których każdy jest projekcją tworzonej sceny na płaszczyznę ekranu, ale rzutowaną z innego punktu. Odległość między tymi punktami jest równa rozstępowi oczu. Każdy z obrazów przeznaczony jest tylko dla jednego oka, a zatem należy wprowadzić mechanizm separacji obrazów, umożliwiający obserwację pierwszego obrazu tylko lewym okiem, a drugiego — tylko prawym. Do separacji obrazów wykorzystywane są różne techniki.

Najprostsze rozwiązanie to zamocowanie stereoskopu na ekranie monitora. W metodzie anaglifowej natomiast, w której każdy z obrazów wyświetlany jest innym kolorem, użytkownik obserwuje tworzoną na ekranie scenę przez specjalne okulary; poszczególne szkła pełnią tu rolę filtrów barwnych. Zastosowanie koloru do separacji obrazów wyklucza jednak możliwość obserwacji scen barwnych.

W rastrowych systemach graficznych najlepsze rezultaty można uzyskać wykorzystując technikę międzyliniowego tworzenia rastra i specjalne okulary polaryzacyjne [5]. Dla każdego oka przeznaczony jest tu jeden półobraz (przy wybieraniu międzyliniowym każdy kadr składa się z dwóch półobrazów). Szkła okularów składają się z dwóch polaryzatorów i umieszczonej między nimi warstwy zawierającej elektrody, których zadaniem jest wytwarzanie pola elektromagnetycznego i zmiana kierunku polaryzacji. Elektrody w poszczególnych szkłach sterowane są naprzemiennie (impulsami synchronizującymi) i umożliwiają obserwację każdym okiem tylko jednego półobrazu. Rozwiązanie to pozwala uzyskać pełny efekt widzenia przestrzennego (również scen barwnych).

* * *

Przedstawione zagadnienia obejmują jedynie najważniejsze aspekty percepcji wzrokowej użytkownika systemów graficznych. Istnieje ponadto szereg zjawisk — nie zawsze dostatecznie wyjaśnionych, chociaż stwierdzanych doświadczalnie — takich jak np. złudzenia optyczne, wrażenie kształtu, zjawisko przedłużonego oddziaływania [6], które również odgrywają istotną rolę w postrzeganiu. Skuteczność odbioru zależy także od warunków obserwacji, zmęczenia użytkownika, przy czym w ustalaniu prawidłowych warunków pracy można tu wykorzystać wyniki badań ergonomicznych telewizji.

Zagadnienia percepcji wzrokowej, w kontekście użytkownika systemów graficznych, stają się obecnie przedmiotem licznych badań. Ich wyniki pozwalają doskonalić ten obecnie najbardziej efektywny sposób dialogu człowieka z komputerem.

LITERATURA

[1] Dorner W. G.: Grafishe Presentation numerisher Daten. Elektronik, nr 25/26, 1981

[2] Frei W.: Image enhancement by histogram hyperbolization Computer Graphics and Image Processing, nr 6, 1977

[3] Nakatani H. i in.: A binocular stereoscopic display system for echocardiography. IEEE Transactions on Biomedical Engineering, vol. BME-26, nr 1, 1979

[4] Reuss J. L., Kertesz A. E.: Microcomputer generation of dynamic stereo graphics for clinical use. IEEE Transactions on Biomedical Engineering vol. BME-28, nr 1, 1981

[5] Roese J. A., McCleary L. E.: Stereoscopic computer graphics for simulation and modeling. Computer Graphics vol. 13, nr 2, 1979

[6] Rosenfeld A., Kak A. C.: Digital Picture Processing. Academic Press, New York, 1976

[7] Walczyk J.: Podstawy systemów telewizji użytkowej. WKiŁ, Warszawa, 1982.

KONFERENCJE

RELCOMEX'84

III Międzynarodowa Konferencja Naukowo-Techniczna „Niezawodność i eksploatacja systemów komputerowych RELCOMEX'84" odbędzie się w maju 1984 w Zamku Książ. (w pobliżu Wałbrzycha). Organizatorami są: SEP, ELWRO, Politechnika Wrocławska i Uniwersytet Wrocławski.

Celem konferencji będzie wymiana informacji o dorobku naukowym w zakresie niezawodności i eksploatacji systemów komputerowych oraz prezentacja wyników praktycznych wdrożeń. Nastąpi podział na sekcje tematyczne:

— Modele matematyczne niezawodności
— Niezawodność i eksploatacja systemów cyfrowych
— Niezawodność i eksploatacja systemów mikrokomputerowych i mikroprocesorów
— Testowanie i diagnostyka układów logicznych i systemów cyfrowych
— Praktyczne aspekty sterowania niezawodnością i eksploatacją systemów cyfrowych
— Niezawodność i technologia przetwarzania w systemach komputerowych.

Pełne teksty wygłaszanych referatów będą publikowane w materiałach konferencyjnych w języku angielskim lub rosyjskim (zgodnie z nadesłanym tekstem).

Szczegółowe informacje o konferencji można uzyskać pod adresem:

RELCOMEX'84 Politechnika Wrocławska
Instytut Cybernetyki Technicznej
ul. Janiszewskiego 11/17
50-372 Wrocław
doc. Wojciech Zamojski
teleks 0712254 pwr, 0712559 pwr, tel. 203-694.

W rozpoczętym dwa lata temu cyklu poświęconym językom programowania i systemom operacyjnym przedstawiliśmy jak dotąd ADĘ, LOGLAN, CP/M i BASIC. W bieżącym numerze proponujemy lekturę przeglądu języków symulacyjnych. Jest to swoista klasa języków przeznaczonych zarówno do symulacji procesów ciągłych, jak i dyskretnych. Choć każdy z tych języków jest na swój sposób interesujący, większość z nich jest dobrze znana i nie miałoby raczej sensu przypominanie opublikowanych po polsku podręczników. Natomiast porównanie podstawowych właściwości języków symulacyjnych jest z pewnością pożyteczne, mimo że — będąc przeznaczone do ściśle określonego celu — nie mają one tak nowoczesnej budowy, jak ADA czy LOGLAN.

W kolejnych numerach zamierzamy przedstawić następne języki programowania: Przemysłowy FORTRAN Czasu Rzeczywistego (IRT-FORTRAN), CHILL, MODULA-2, FORTH, C, BCPL i PEARL oraz system operacyjny UNIX.

Redakcja

TERESA WILCZEK
Centrum Obliczeniowe SGGW-AR
Warszawa

Aspekty użytkowe języków symulacyjnych (1)

Symulacja dyskretna

Zastosowanie maszyn cyfrowych do badań systemów rzeczywistych spowodowało powstanie specjalizowanych języków i pakietów programowania. Pozwalają one na budowę programów symulacyjnych opisujących reguły funkcjonowania bądź potencjalne możliwości działania wyróżnionych systemów.

Celem artykułu jest dokonanie przeglądu i oceny wartości użytkowych języków i pakietów symulacyjnych stosowanych w kraju. Nie jest to problem banalny. Można doskonale znać konkretne języki i pakiety symulacyjne, ale — ze względu na nieznajomość cech zaleceń ogólnych — używać tych narzędzi niewłaściwie. Z drugiej strony zbiór pojęć języka symulacyjnego narzuca sposób odwzorowania badanego systemu, a możliwości manipulacji zbudowanym modelem symulacyjnym zawężają się do tych, które zawarto w danym języku. Wpływ języka symulacyjnego waha się od subtelnego oddziaływania na strukturę modelu — aż do narzucania ostrych ograniczeń na jego możliwości interpretacyjne.

Mgr inż. TERESA WILCZEK ukończyła w 1979 r. studia na Wydziale Elekroniki Politechniki Warszawskiej. Pracę zawodową rozpoczęła w Instytucie Badań Systemowych PAN w Samodzielnej Pracowni Modelowania Systemów Transportu Kolejowego. Od 1981 r. pracuje w Centrum Obliczeniowym Akademii Rolniczej w Warszawie. Zajmuje się oprogramowaniem w dziedzinie statystyki matematycznej oraz teorią eksperymentu.

Porównanie efektywności i ocena jakości rozmaitych języków symulacyjnych sprawia poważne trudności, rzadko bowiem można znaleźć takie ich cechy, które są porównywalne w jednoznacznych miernikach. Istnieje jednak ogólna klasyfikacja języków symulacyjnych, a w ramach danej klasy, kierując się doświadczeniem i wiedzą ogólną, można określić cechy każdego języka decydujące o jego aspektach użytkowych.

KLASYFIKACJA JĘZYKÓW (PAKIETÓW) SYMULACYJNYCH

Metody modelowania cyfrowego pozwalają wyodrębnić trzy klasy języków symulacyjnych:

- języki (pakiety) do symulacji systemów ciągłych (symulacja ciągła)
- języki (pakiety) do symulacji systemów dyskretnych (symulacja dyskretna)
- języki do symulacji systemów ciągło-dyskretnych (symulacja mieszana).

Pierwszą decyzją na etapie poszukiwania narzędzia symulacji jest wybór metody modelowania, a więc wybór jednej z klas języków symulacyjnych. Decyzja ta podejmowana jest w oparciu o analizę własności modelowanego systemu. Należy tu zwrócić uwagę na pewien istotny fakt — opis badanego systemu jest modelem abstrakcyjnym jego rzeczywistej struktury. Z tego względu sposób opisu nie zawsze musi odpowiadać typowi rozważanego systemu. Wybierając określoną metodę symulacji należy uwzględnić zarówno cechy systemu, jak również mieć na uwadze cel badania symulacyjnego.

Przegląd języków symulacyjnych wykazuje największe zróżnicowanie klasy języków do symulacji dyskretnej. Metoda symulacji dyskretnej jest jedną z historycznie najwcześniej, a obecnie — najczęściej, stosowanych metod symulacji cyfrowej. Najnowsza zaś, najmniej rozbudowana i nie omówiona jeszcze w wydawnictwach krajowych jest klasa języków do symulacji ciągło-dyskretnej, a więc i metoda symulacji mieszanej.

SYMULACJA SYSTEMÓW DYSKRETNYCH

Jednym z kryteriów ułatwiających wybór języka (pakietu) do symulacji dyskretnej jest — stosowany w modelowaniu symulacyjnym — podział obiektów systemu na czynne i bierne. Obiekty czynne mogą zmieniać stan lub właściwości innych obiektów, natomiast obiekty bierne zmieniają się pod wpływem niezależnych od nich oddziaływań. Analogicznie do procesów przemysłowych, można podzielić obiekty na „materiały", poddawane obróbce, oraz „maszyny", dokonujące obróbki. Jeśli — na przykład — rozpatruje się zagadnienie produkcji masowej określonego wyrobu, przy użyciu skomplikowanego ciągu produkcyjnego, złożonego z wielu maszyn, dominującą rolę odgrywają w modelu obiekty czynne (maszyny). Jeśli natomiast rozpatruje się proces, w którym występuje wiele materiałów (obiektów biernych), a sposób ich przetwarzania zależy od ich indywidualnych właściwości, korzystniejszy okazuje się opis procesu ukierunkowany właśnie na materiały.

Klasy języków

Podstawowe pojęcia używane przy programowaniu symulacji dyskretnej to:

- zdarzenie
- działanie
- proces

Zdarzenie definiowane jest — ogólnie rzecz biorąc — jako zmiana stanu obiektu w danej chwili.

Metody wyboru działań

W metodzie wybierania działań posługujemy się wykazem wszystkich możliwych działań. Przy każdym zdarzeniu przegląda się cały wykaz działań, po kolei wypróbowując, czy mogą one nastąpić w związku ze stanem każdego z obiektów systemu. Po przejrzeniu całego wykazu, staje przed nami problem działań uzależnionych od warunków. Najłatwiejszy sposób polega na ponownym przejrzeniu całego wykazu działań; przeglądanie powtarza się dopóty, dopóki dla danego stanu „zegara symulacji" nie może nastąpić żadne działanie.

W praktyce stosuje się również tzw. strukturę trójfazową. W strukturze tej rozróżnia się działania uzależnione od czasu (tj. takie, które następują jedynie na skutek samgo upływu czasu) i działania uzależnione od warunków. Czystą metodę przeglądania działań łatwo jest sformułować i zaprogramować, płacimy jednak za to długim czasem realizacji obliczeń na maszynie cyfrowej. Metoda trójfazowa jest wprawdzie nieco trudniejsza w oprogramowaniu, niemniej jest efektywniejsza. Czystą metodę wyboru działań zastosowano w języku CSL, zaś metoda trójfazowa jest realizowana w pakiecie symulacyjnym SIMON.

CSL [3] jest językiem symulacyjnym ogólnego przeznaczenia opartym na FORTRANIE. Konstrukcja języka umożliwia programującemu poświęcenie więcej uwagi i czasu na metodę rozwiązywania problemu, a mniej na samą technikę programowania. Program napisany w CSL opisuje szereg działań. Każde z nich dotyczy zbioru czynności zmieniających stan modelu. Tak więc program CSL może być użyty do sterowania bieżącym ciągiem operacji lub do podejmowania decyzji na najbliższą przyszłość. CSL jest językiem polecanym do modelowania systemów, w których dominującą rolę odgrywają obiekty czynne. Nie będziemy omawiać cech tego języka ani założeń konstrukcji, jednak w celu zilustrowania jego koncepcji przytoczymy krótki przykład.

```
EXISTS K LISTA ≥ 200
     OSOBA.K IN OBSZAR.3
     WIEK(K) LT 45
     CHAIN
          ODLEGLOSC(K) GT 500
          OR KARALNOSC(K) LT 2
```

Przedstawiony układ instrukcji można zinterpretować następująco: jeżeli w zbiorze LISTA jest zarejestrowana co najmniej jedna osoba, która mieszka w „obszarze 3", jej wiek nie przekracza 45 lat oraz przebywa obecnie w odległości większej niż 500 km lub była karana sądownie najwyżej jeden raz, to wynik badania jest pozytywny i sterowanie przechodzi do kolejnej instrukcji. Zmienna K służy w tym przykładzie jako identyfikator k-tego elementu klasy OSOBA, który może należeć do zbioru LISTA i OBSZAR.+3, a także jako indeks tablicy ODLEGŁOŚĆ i KARALNOSC.

Podobną technikę modelowania można zrezalizować za pomocą pakietu SIMON [7], zawierającego szereg procedur symulacyjnych napisanych w języku ALGOL. Koncepcja trójfazowości zwiększa w tym przypadku szybkość realizacji eksperymentu symulacyjnego. Należy zaznaczyć, że SIMON w zasadzie nie posiada aparatu zbierania i obróbki wyników symulacji z wyjątkiem konstrukcji histogramów. SIMON dostosowany jest do symulacji systemów o dominujących obiektach czynnych.

Metoda planowania zdarzenia

Przy stosowaniu tej metody jednostką symulacji jest zbiór reguł związanych nie z działaniem, lecz z całym kompleksem działań, które mogą wystąpić jako jedno złożone zdarzenie. Zamiast upewniać się, czy wszystkie działania uzależnione od warunków będą wykonane, przeglądając je od nowa, formułuje się tu konkretne instrukcje, które bezpośrednio kierują komputer na istotne działanie. Programowanie symulacyjne przy użyciu metody następstwa zdarzeń bywa skomplikowane, zwłaszcza gdy symulujemy systemy bardziej złożone. Osiągamy za to znacznie większą wydajność niż w przypadku metody przeglądania działań.

Językiem o strukturze opartej na opisanej koncepcji jest SIMSCRIPT [1, 6], zbudowany w oparciu o FORTRAN. Kładzie on nacisk na modułowy sposób zapisywania programów. Każde zdarzenie jest wyodrębnioną procedurą. Z punktu widzenia czasochłonności procesu symulacji na maszynie cyfrowej, SIMSCRIPT jest językiem stosunkowo efektywnym, tzn. czas realizacji programu napisanego w SIMSCRIPCIE jest krótszy niż realizacja analogicznego modelu symulacyjnego np. w omawianym dalej języku GPSS. SIMSCRIPT nadaje się do rozwiązywania wszelkich problemów związanych z obsługą masową, tzn. takich systemów, w których istotną rolę odgrywają trzy następujące zagadnienia:

- zagadnienie sposobu obsługi zgłoszeń (zapotrzebowań na obsługę), które obejmuje częstość zgłoszeń, liczbę zadań do wykonania przy jednym zgłoszeniu oraz wymagane środki i czas obsługi jednego zadania
- problem przydziału środków na stanowiska obsługi i na poszczególne zadania
- problem wybierania zadań do obsługi.

SIMSCRIPT polecany jest do modelowania systemów, w których dominującą rolę odgrywają obiekty bierne. Na przykład — wyszukanie i usunięcie ze zbioru KOLEJKA pierwszego obiektu INFORMACJA z atrybutem PRIORYTET równym 5 można w języku SIMSCRIPT zapisać następująco:

```
FOR EVERY INFORMACJA WITH PRIORYTET=5,
     FIND THE FIRST CASE
IF FOUND REMOVE THIS INFORMACJA FROM KOLEJKA
```

Jak widać, język ten ma składnię i cechy zbliżone do zdań języka naturalnego, co jest środkiem usprawniającym konstrukcję modelu.

Metoda przebiegu procesów

Koncepcja przebiegu procesów wykorzystuje takie terminy, jak: „czekanie do chwili..." (termin warunkowy) oraz „przesunąć..." (termin bezwarunkowy). Modelowany system reprezentowany jest przez zbiór współbieżnych, współdziałających procesów. Metoda przebiegu procesów jest, ogólnie rzecz biorąc, najbardziej polecana przy projektowaniu programów symulacyjnych.

Metoda ta pozwala lepiej niż poprzednie metody zrozumieć przebieg modelowanej pracy systemu, wymaga jednak dużej precyzji przy jej formalizacji. Wynika to z konieczności odpowiedniego opisu stanów obiektów występujących w obsłudze, jak też z przyjętej tu specyfiki kolejek. Językami opartymi na koncepcji interakcji procesów, są: SIMULA, GPSS oraz pakiet GPSS-F.

Tabela 1. Podstawowe pojęcie używane w procesie budowy modelu w poszczególnych językach symulacyjnych

	SIMSCRIPT	CSL	SIMON	SIMULA	GPSS	GPSS-F
Indentyfikacja obiektów i atrybutów	obiekt, atrybut	obiekt, właściwość	obiekt, właściwość	proces, atrybut	transakcja (zadanie), parametr	transakcja (zadanie), parametr
Powiązania między obiektami	zbiór	zbiór	zbiór	zbiór	łańcuch	łańcuch
Metody generowania biektów	wygenerować nowy obiekt	nie istnieją	nie istnieją	wygenerować nowy obiekt	wygenerować nową transakcję (zadanie)	wygenerować nową transakcję (zadanie)

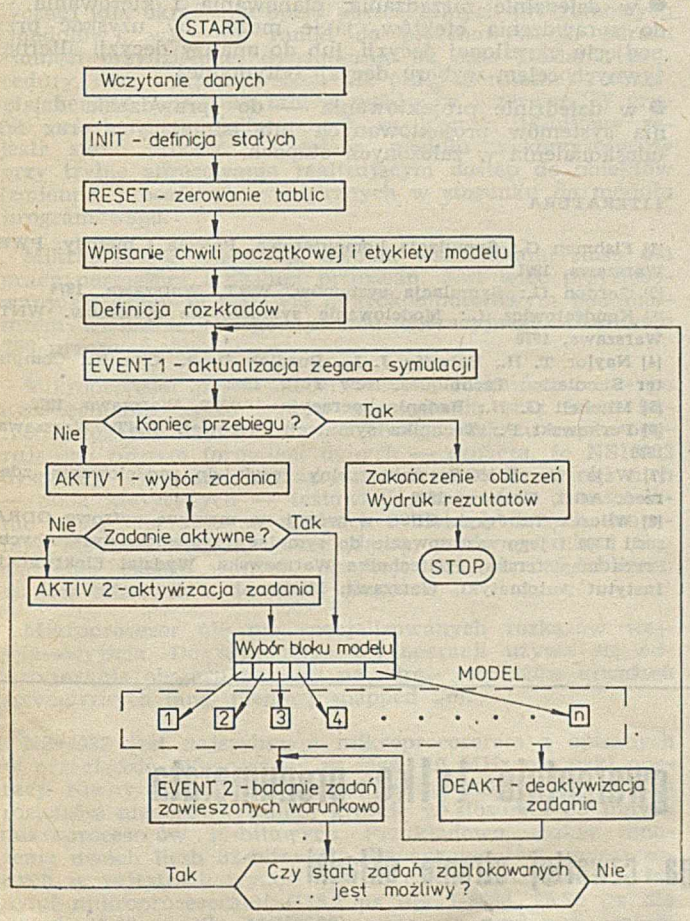
SIMULA [1, 6] została zbudowana na bazie ALGOLU i zawiera ten język jako podzbiór. Podstawowym pojęciem modelowania w SIMULI jest proces. Język ten ma rozbudowany aparat operacji, umożliwiających porządkowanie zbiorów, wyszukiwanie elementów według zadanych kryteriów, badanie zawartości zbiorów itp. Dopuszcza się tu zarówno pasywny, jak i aktywny charakter wszelkich elementów systemu.

SIMULA charakteryzuje się dużą elastycznością, dostarcza ona bowiem szereg terminów związanych z operacjami dynamicznymi lub logicznymi — i dopiero z nich buduje się procedury opisujące działanie każdego elementu systemu. Fakt ten decyduje o uniwersalności tego języka, z drugiej jednak strony opanowanie go jest trudniejsze niż nauczenie się bardziej specjalizowanych symulatorów, jak np. opisane już SIMSCRIPT czy CSL.

Sekwencja operacji SIMULI:

```
begin ref (PROCESS)A;
    . . .
    A = new KLENT;
    activate A at T;
    . . .
end;
```

powoduje włączenie elementu A reprezentującego klasę procesów KLENT do harmonogramu zdarzeń z czasem uaktywnienia danego egzemplarza procesu równym T.



GPSS — F — sterowanie dynamiką modelu

Tabela 2. Zestawienie języków symulacyjnych z punktu widzenia ich przydatności do modelowania różnych systemów

	Typ obiektu dominującego w systemie	Zalecenia
SIMSCRIPT	obiekty bierne	systemy masowej obsługi
CSL	obiekty czynne	systemy masowej obsługi wszelkie systemy dyskretne
SIMON	obiekty czynne	systemy masowej obsługi
SIMULA	obiekty czynne i bierne	złożone systemy decyzyjne
GPSS	obiekty bierne	systemy masowej obsługi
GPSS-F	obiekty bierne	systemy masowej obsługi

Językiem realizującym koncepcję procesów jest również GPSS [2, 6]. Specyficzną cechą GPSS, różniącą go od innych języków do symulacji systemów dyskretnych, jest to, że model symulacyjny badanego systemu, zbudowany w oparciu o ten język, ma strukturę blokową. Język ten dostarcza niezmiennego i ściśle zdefiniowanego zestawu bloków o ustalonych regułach działania.

Dominującą rolę w GPSS odgrywają obiekty bierne, a jedynymi obiektami czynnymi są występujące w systemie tzw. zadania (TRANSACTIONS). Zadania mogą reprezentować elementy na linii produkcyjnej, statki przypływające do portu, klientów w zakładzie usługowym itp. Blokowe oprogramowanie wszystkich typowych sytuacji spotykanych w modelach systemów dyskretnych oraz łatwa do opanowania logika GPSS czynią go narzędziem szczególnie cennym. Językiem tym mogą posługiwać się nawet te osoby, które mają niewielkie doświadczenie w programowaniu. Należy podkreślić, że budowa i terminologia języka GPSS jest najbliższa terminologii systemów masowej obsługi.

Przykładowy ciąg instrukcji:

```
QUEUE 2
SEIZE 5
DEPART 2
ADVANCE 45
RELEASE 5
```

realizuje następujące czynności: ustawienie się zadania w kolejce o numerze 2, zajęcie urządzenia numer 5 (o ile urządzenie to jest wolne), wyjście z kolejki zadań czekających na obsługę w tym urządzeniu (DEPART), obsługę (czas obsługi równy 45 jednostek) oraz zwolnienie urządzenia (RELEASE).

Pakiet GPSS-F [8] stanowi natomiast zbiór procedur napisanych w języku FORTRAN. Możliwości interpretacyjne i struktura GPSS-F łączy zalety języka GPSS z możliwościami wynikającymi z bezpośredniego dostępu do procedur opisujących poszczególne bloki. Użytkownik korzystający z GPSS-F powinien poznać i zaprogramować fragment realizujący dynamikę modelu oraz strukturę systemu. Część opisująca strukturę jest odpowiednikiem programu napisanego w języku GPSS.

Dla przykładu — realizacja przytoczonego w GPSS fragmentu modelu, w GPSS-F wygląda następująco:

CALL QUEUE (2, 1, 1) — wejście jednego zadania do kolejki nr 2, żądanie wydruku komunikatów kontrolnych

Tabela 3. Cechy języków do symulacji dyskretnej z użytkowego punktu widzenia

	Aparat generowania liczb pseudolosowych	Aparat analizy statystycznej	Aparat tworzenia wykresów (np. histogramy)
SIMSCRIPT	rozkład równomierny	istnieje	nie istnieje
CSL	rozkład równomierny, normalny, Poissona, wykładniczy	istnieje	nie istnieje
SIMON	rozkład równomierny	istnieje	istnieje
SIMULA	rozkład równomierny, Poissona, Erlanga, wykładniczy, normalny	istnieje	nie istnieje
GPSS	rozkład równomierny	istnieje	istnieje
GPSS-F	rozkład równomierny, wykładniczy, normalny, Poissona	istnieje	istnieje

3 CALL SEIZE (5, 3, KL, 1) — zajęcie urządzenia nr 5, wydruk komunikatów kontrolnych

CALL DEPART (2, 1, 0) — wyjście jednego zadania z kolejki nr 2, bez wydruku komunikatów kontrolnych

7 CALL ADVANC (45, 0, 7, RN1, KL, 1) — modelowanie czasu obsługi w urządzeniu

CALL RELEAS (5, 0, KL) — zwolnienie urządzenia nr 5, bez wydruku komunikatów kontrolnych (parametr KL jest tzw. indykatorem błędu)

Procedury sterujące dynamiką modelu realizują następujące czynności (rys.):

- ustalenie wielkości pomocniczych (zerowanie tablic, definiowanie stałych)
- wyszukanie następnego zdarzenia w procesie symulacji
- wybór działania i sprawdzenie warunków
- aktywizacja i deaktywizacja zadań.

Pakiet procedur GPSS-F stanowił dla producentów podstawę przy budowie tzw. wersji symbolicznej języka GPSS. Tworzenie wersji symbolicznej polega na budowie translatora tłumaczącego nazwy symboliczne bloków na procedury bazy języka, napisane w FORTRANIE. Wersja podstawowa nie wymaga dodatkowego translatora, ponieważ procedury bazy języka wywoływane są jawnie. Jawność instrukcji symulacyjnych nie stawia ograniczeń co do liczby bloków i atrybutów numerycznych generowanych zadań, pozwala ponadto rozszerzyć możliwości pakietu przez definiowanie nowych bloków, rozkładów statystycznych itd. Znając strukturę pakietu, programista może określać nowe procedury, nie tylko rozszerzające, ale również różnicujące zastosowanie pakietu.

PORÓWNANIE JĘZYKÓW SYMULACJI DYSKRETNEJ

Jak już wspomniano, stworzenie jednoznacznych kryteriów porównawczych dla oceny wartości funkcjonalnej języków symulacyjnych nie jest możliwe. Przydatna jest natomiast znajomość użytkowej charakterystyki tych języków, będącej jednym z kryteriów wyboru narzędzia modelowania. W tabeli 1 zebrano podstawowe pojęcia używane w procesie budowy modelu w poszczególnych językach symulacyjnych.

W tabeli 2 podano natomiast zestawienie języków pod kątem ich przydatności do modelowania różnego rodzaju systemów.

Według ankiet przeprowadzonych w Stanach Zjednoczonych, GPSS jest oceniany jako język najłatwiejszy w użyciu. SIMULA natomiast jest jednym z języków trudniejszych, zapewnia jednak odwzorowanie najszerszej gamy problemów. Ankietowani użytkownicy najczęściej posługiwali się SIMSCRIPTEM. Wynika to z jego efektywności obliczeniowej, modułowego zapisu modeli, a także dzięki utrzymaniu odpowiedniego kompromisu między wysiłkiem włożonym w nauczenie się języka a jego możliwościami programowymi. GPSS, mimo łatwości w programowaniu, narzuca największe ograniczenia w stosunku do budowanych modeli.

Zestawienie cech języków do symulacji dyskretnej z czysto użytkowego punktu widzenia zawiera tabela 3.

Przy wyborze języka symulacyjnego użytkownik jest często ograniczony typem dostępnej maszyny cyfrowej oraz jej oprogramowaniem. Należy w tym przypadku zwrócić uwagę na przydatność pakietów symulacyjnych typu SIMON czy GPSS-F. Pakiety te można bowiem niewielkim nakładem kosztów zainstalować na każdej maszynie, która dysponuje językiem oprogramowania danego pakietu.

* * *

Na zakończenie przedstawionych rozważań należy zwrócić uwagę na jeszcze jeden istotny szczegół, a mianowicie ten, że symulacja komputerowa, tak jak wszelkie inne dziedziny programowania, jest sztuką i to sztuką nie tylko programowania, lecz również tworzenia modeli abstrakcyjnych określanych przez konkretny cel badań symulacyjnych. Wymaga ona nie tylko znajomości określonych przepisów i reguł, ale również intuicji wynikającej z umiejętności abstrakcyjnego myślenia oraz — praktyki. Właściwe wykorzystanie tej sztuki pozwala „laboratoryjnie" prowadzić różne eksperymenty, np.:

- w dziedzinie zarządzania, planowania i kierowania — do sprawdzenia efektów, jakie można by uzyskać przy podjęciu określonej decyzji, lub do analizy decyzji alternatywnych celem wyboru decyzji optymalnej
- w dziedzinie projektowania — do sprawdzenia działania systemów projektowanych lub istniejących oraz ich udoskonalenia w założonych etapach.

LITERATURA

[1] Fishman G.: Symulacja komputerowa. Pojęcia i metody. PWE, Warszawa, 1981
[2] Gordon G.: Symulacja systemów. WNT, Warszawa, 1974
[3] Kondratowicz L.: Modelowanie symulacyjne systemów. WNT, Warszawa, 1978
[4] Naylor T. H., Balmffy J. L., Burdick D. S., Chu K.: Computer Simulation Techniques. New York, 1966
[5] Mitchell G. H.: Badania operacyjne. WNT, Warszawa, 1977
[6] Perkowski P.: Technika symulacji cyfrowej. WNT, Warszawa, 1980
[7] Wajs W.: SIMON, uniwersalny język do modelowania zdarzeń. AGH, Kraków, 1980
[8] Wilczek T.: Język GPSS w wersji na maszynę cyfrową ODRA serii 1300 i jego zastosowanie do symulacji systemów dyskretnych. Praca magisterska, Politechnika Warszawska, Wydział Elektroniki, Instytut Automatyki, Warszawa, 1979.

Stały kontakt z INFORMATYKĄ gwarantuje tylko prenumerata

Zasady jej zamawiania — na czwartej stronie okładki

JERZY GRABOWSKI
Warszawa

Przegląd mikroprocesorów 16-bitowych (6)
NATIONAL SEMICONDUCTOR 16000

Mikroprocesory z rodziny NS16000 są najbardziej złożonymi układami z opisanych w całym przeglądzie. Rodzina NS16000 obejmuje (stan z roku 1980) dwa mikroprocesory: 16016 i 16032, z których tylko drugi był dostępny na rynku w 1980 r. Niniejszy opis będzie dotyczył właśnie mikroprocesora NS16032.

Mikroprocesor ten ma 24-bitowy licznik rozkazów i inne rejestry adresowe o tej pojemności, ale — podobnie jak MC68000 — architekturę w pełni 32-bitową. Może więc aktualnie odwoływać się bezpośrednio do 16 M bajtów pamięci. Ma on osiem 32-bitowych rejestrów uniwersalnych stosowanych przy operacjach na danych oraz do adresowania pamięci, przy bardzo rozbudowanym i wyrafinowanym zestawie trybów adresowania. Można przy ich użyciu wykonywać działania na danych bajtowych, słowach pojedynczych, podwójnych (32 bity) i poczwórnych (64 bity).

Rejestry adresowe są 24-bitowe i należą do grupy rejestrów o funkcjach specjalnych. Jest wśród nich licznik rozkazów i rejestry wskaźników adresowych: bazy statycznej (dla określenia opisu poszczególnych programowych „modułów"), wskaźnik ramki do przechowywania adresu pamięci przydzielonej dynamicznie na czas działania procedury, dwa wskaźniki stosu dla pracy normalnej i obsługi przerwań, wskaźnik tablicy wektorów przerwań. W grupie rejestrów specjalizowanych są jeszcze: 16-bitowy rejestr stanu procesora i rejestr modułu wykorzystywany przy trybie adresowania realizującym dostęp do obiektów (zmienne, procedury) zewnętrznych w stosunku do modułu programowego.

Mikroprocesor dysponuje dwoma wskaźnikami stosu dla pracy normalnej i obsługi przerwań. Jest więc przystosowany do pracy w zestawie mikrokomputerowym pod nadzorem wielozadaniowego systemu operacyjnego. Obsługuje 256 przerwań.

Mikroprocesor ma 81 podstawowych rozkazów, co — po uwzględnieniu ośmiu trybów adresowania oraz faktu istnienia różnych wariantów tego samego rozkazu często operującego różnym formatem danych — oznacza, że NS16032 dysponuje tysiącami znaczących rozkazów. Są wśród nich — obok klasycznych — testowanie, kasowanie i ustawianie bitów, operacje na napisach i blokach danych, transfery blokowe, rozkazy przeglądania, porównywania i tłumaczenia ciągów danych, operacje działań na tablicach ciągów bitowych o dowolnej długości.

Mikroprocesor nie ma specjalizowanych rozkazów wejścia—wyjścia. Do wykonania tej operacji używa się odwzorowania obszaru pamięci na adresy rejestrów urządzeń zewnętrznych (ang. memory mapped i/o).

NS16032 jest najszybszym mikroprocesorem z opisanych w przeglądzie. Wyposażony w zegar 10 MHz ma cykl operacji równy 100 ns. Zdumiewające są czasy wykonywania rozkazów nieporównywalnie krótkie w stosunku do innych mikroprocesorów 16-bitowych. Przykładowo, rozkaz mnożenia dwóch liczb 32-bitowych dla argumentów umieszczonych w rejestrach wynosi 7,6 μs, co w zestawieniu z innymi mikroprocesorami (115,2 μs dla I-8086, 85,75 μs dla Z8000, 43,00 μs dla MC68000) może się wydawać czasem szokującym. W tabeli porównano typowe czasy wykonywania także dla innych rozkazów INTEL 8086, Z8000, MC68000 i NS16000, dając czytelnikowi rozeznanie w szybkości działania tego ostatniego.

Ciekawą właściwością NS16000 jest możliwość wykonywania programu sformułowanego dla ośmiobitowego mikroprocesora INTEL 8080/8085. Kod I-8080 nie jest interpretowany, lecz bezpośrednio wykonywany przez sprzęt NS16000.

Firma NATIONAL SEMICONDUCTOR oferuje dla systemów mikrokomputerowych wyposażonych w jednostkę centralną NS16000 różne układy pomocnicze, np. dla zarządzania pamięcią, sterowniki przerwań, zegary, a także układy wykonywania operacji zmiennoprzecinkowych. Te ostatnie są wbudowane w oddzielny mikroprocesor pełniący w systemie mikrokomputerowym rolę procesora podrzędnego.

Porównanie czasów wykonywania niektórych rozkazów dla mikroprocesorów I-8086, Z8000, MC68000 I NS16000

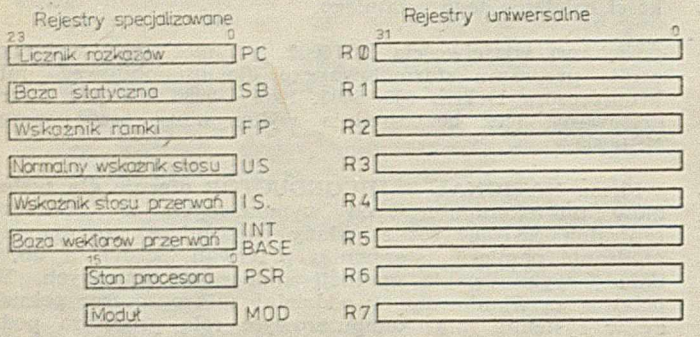
Operacja	Typowy czas wykonania μs				
	Dane	8086	Z8000	MC68000	NS16032
MOV R, R (prześlij zawartość rejestru)	B, W	0,40	0,75	0,50	0,30
	D	0,80	1,25	0,50	0,30
MOV M, R (prześlij zawartość pamięci do rejestru)	B, W	3,40	3,50	1,50	1,00
	D	6,80	4,25	2,00	1,40
MOV M, M (prześlij zawartość pamięci)	B, W	7,00	7,00	2,50	1,60
	D	14,00	8,50	3,75	2,40
ADD M, R (dodaj zawartość pamięci do rejestru)	B, W	3,60	3,75	1,50	1,10
	D	7,20	5,25	2,25	1,50
CMP M, M (porównaj zawartości pamięci)	B, W	7,00	7,25	3,00	1,80
	D	14,00	9,50	4,00	2,60
MULT R, R (pomnóż zawartości rejestrów)	B	13,00	20,25	—	2,80
	W	23,60	16,00	8,25	4,40
	D	15,20	85,75	43,00	7,60
BCC (skok warunkowy)	tak	1,60	1,50	1,25	1,40
	nie	0,80	1,50	1,00	0,70
ACBI (modyfikuj rejestr i skocz gdy nie zero)	tak	2,20	2,75	1,25	1,30
BSR (wywołanie podprogramu)		3,80	3,75	2,25	2,50
		5 MHz	4 MHz	8 MHz	10 MHz

Oznaczenia: B — bajt, W — słowo 16-bitowe, D — podwójne słowo; tak, nie — skok efektywny, nieefektywny

Ze względu na funkcjonalne właściwości NS16032, jego opis wprowadza szereg pojęć nowych dla programistów komputerów i wart jest przestudiowania choćby dla spojrzenia w dziedzinę rozwiązań, które wydają się być przyszłościowe.

REJESTRY NS16032

Rejestry mikroprocesora NS16032 przedstawiono na rysunku 1. Rejestry uniwersalne 32-bitowe oznaczono jako R0, R1,..., R7. Mogą one przechowywać i wykorzystywać w operacjach dane 8-, 16-, 32- i 64-bitowe. Argumenty 64-bitowe są wykorzystywane tylko w rozkazach mnożenia i dzielenia, ale pozostałe typy danych mogą być wybierane wariantowo dla każdego rozkazu arytmetycznego, logicznego, przesłania i działania na ciągach danych. W operacjach bajtowych bardziej znaczące bajty rejestru nie są osiągalne dla operacji działającej tylko na najmniej znaczącym bajcie. Pozostałe bajty rejestru można jednak wykorzystywać do przechowywania wyników pośrednich. Analogiczne ograniczenie nie wstępuje dla operacji na danych 16-bitowych, gdyż obydwie połówki rejestru mogą być stosowane zarówno jako argumenty, jak i wynik działania rozkazu.



Rys. 1. Rejestry mikroprocesora NS16032

Licznik rozkazów PC jest rejestrem 24-bitowym, ale — podobnie jak w przypadku MC68000 — jest zrealizowany wewnętrznie jako 32-bitowy. Umożliwia to rozszerzenie zakresu adresowego do 8092 M bajtów pamięci! Także inne rejestry specjalizowane mają budowę 32-bitową. Projektant mikroprocesora nie zawarł w układzie obsługi segmentacji pamięci, lecz istnieją układy umożliwiające jej podział i ochronę. Bez układów zarządzania pamięcią gigantyczny obszar adresowy procesora zdaje się być raczej wadą niż zaletą, gdyż z jednej strony trudno sobie wyobrazić rozsądne wykorzystanie tak dużej pamięci przez jedno zadanie, jak z drugiej strony — skuteczną ochronę systemową wielu zadań współbieżnych.

Rejestry US i IA (ang. user stack, interrupt stack) są wskaźnikami „stosu użytkownika" i „stosu przerwań" przeznaczanymi odpowiednio dla pracy normalnej i systemowej.

Rejestr INT BASE (ang. interrupt base) wskazuje obszar pamięci przechowującej wektory przerwań. Obszar ten może być umieszczony w dowolnym miejscu zakresu adresowego. Tablica wektorów przerwań ma miejsce na 256 elementów.

Rejestr wskaźnika ramki FP (ang. frame pointer) jest używany do przechowywania adresu początku obszaru danych lokalnych dla procedury (parametrów aktualnych lub lokalnych zmiennych), który jest rezerwowany dynamicznie na stosie w chwili wywołania procedury i kasowany po powrocie do programu głównego. Funkcjonalnie podobne rozwiązanie jest znane czytelnikowi z lektury artykułu o mikroprocesorze MC68000.

Rejestr stanu PSR jest 16-bitowy i ma zdefiniowane znaczenia dla 10 bitów:

I — blokada przerwań
P — praca krokowa
S — wybór wskaźnika stosu SP0 lub SP1
U — tryb pracy systemowy lub użytkowy
N — ujemny wynik operacji arytmetycznej
Z — zerowy wynik operacji arytmetycznej
F — ustawiany przez specjalizowane rozkazy
L — „mniejszy od..." dla wyniku operacji arytmetycznej
T — generowanie przerwań po każdym rozkazie
C — przeniesienie w wyniku operacji arytmetycznej.

Niektóre z tych bitów są dostępne tylko w trybie pracy uprzywilejowanej, inne są tylko zapisywalne.

Specjalnego omówienia wymagają rejestry SB (ang. static base register) i MOD (ang. module register). Są one używane w takim trybie adresowania, który adresy względne modułu przekształca przez zawartości tablic systemowych na adresy bezwzględne. Działanie jednego z trybów adresowania (tryb zewnętrzny, por. „Tryby adresowania") można porównać do transformacji adresów kodu przesuwalnego, dokonywanej w konwencjonalnych rozwiązaniach przez program łączący. W przypadku NS16032 transformacja taka jest realizowana w czasie wykonywania programu, a funkcję programu łączącego przejmuje na siebie sprzęt mikroprocesora.

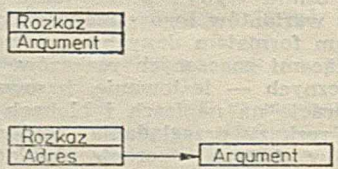
Modułem w sensie właściwości funkcjonalnych NS16000 nazywa się ciąg rozkazów nie używający adresów bezwzględnych. Gdy mikroprocesor chce adresować dane przechowywane w pamięci używa trybu adresowania generującego adres efektywny przez odwołanie do wspomnianych tablic. Tablica taka nazywa się tablicą łączącą. Przechowuje ona 32-bitowe elementy reprezentujące 24-bitowe adresy. Jej początek jest określany zawartością rejestru SB. Z kolei, opis każdego modułu składa się z trzech elementów: (a) — wskaźnika do zmiennych, które są zewnętrzne lub globalne dla modułu; (b) — adresu pierwszego, reprezentującego moduł elementu w tablicy łączącej; (c) — wskaźnika do obszaru kodu rozkazów modułu. Czwarty element opisu modułu jest zarezerwowany dla przyszłych zastosowań.

TRYBY ADRESOWANIA

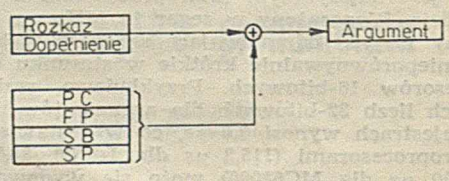
Rozkazy mikroprocesorów NS16000 odwołują się do ośmiu trybów adresowania, które są w pełni symetryczne względem różnych rozkazów i obu ich argumentów. Właściwość ta — bardzo ułatwiająca programowanie — nie była utrzymana w innych mikroprocesorach z wyjątkiem LSI-11. Wśród trybów adresowania, obok klasycznych, są także zupełnie nowe. Oto tryby adresowania mikroprocesorów NS16000:

1) rejestrowy
2) prosty
3) bezwzględny
4) względny rejestrowy
5) względny pamięciowy
6) stosowy
7) zewnętrzny
8) indeksowy skalowany.

Tryby adresowania 1, 2 i 3 nie wymagają oddzielnego komentarza (rys. 2), podobnie jak tryb 4 — względny rejestrowy (rys. 3) — znany z opisu wcześniej przedstawionych mikroprocesorów 16-bitowych. Wówczas jednak mó-

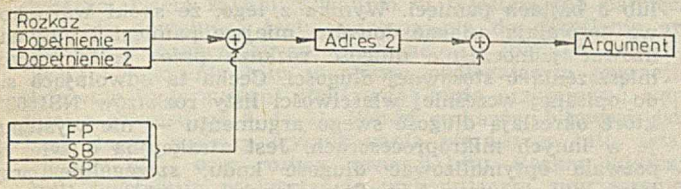


Rys. 2. Tryby adresowania NS16032: rejestrowy, prosty, bezwzględny.
REJESTROWY — argument zawarty w rejestrze uniwersalnym
PROSTY — argument zawarty w kodzie rozkazu
BEZWZGLĘDNY — adres argumentu zawarty w kodzie rozkazu



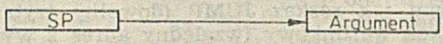
Rys. 3. Tryby adresowania NS16032: względny rejestrowy

wiło się o adresowaniu względem licznika rozkazów, gdyż adres argumentu obliczany był jako suma przemieszczenia i zawartości PC. W rozkazach mikroprocesora NS16000 przemieszczenie może być dodawane także do innych rejestrów: wskaźnika ramki FP, wskaźnika stosu SP i bazy statycznej SB.



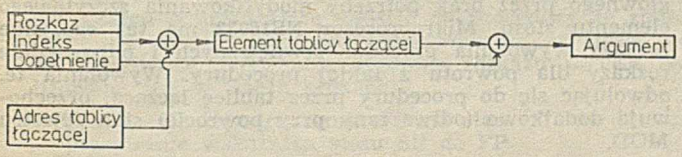
Rys. 4. Tryby adresowania NS16032: względny pamięciowy

Tryb adresowania względnego przez pamięć (tryb 5, rys. 4) używa podwójnego wyrównania i generuje dwa adresy. Pierwsze wyrównanie jest dodawane do zawartości wskaźnika ramki FP, stosu SP lub bazy statycznej SB (w odróżnieniu od trybu 4 nie jest tu dozwolony rejestr PC). Uzyskany w ten sposób pierwszy adres wskazuje komórkę pamięci, której zawartość dodana do drugiego wyrównania określa efektywny adres argumentu.



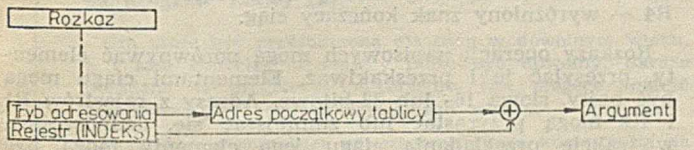
Rys. 5. Tryby adresowania NS16032: stosowy

Wyodrębniony tutaj tryb adresowania stosowego (tryb 6, rys. 5) występował we wcześniej opisywanych mikroprocesorach często w postaci adresowania niejawnego. Potrzeba wyodrębnienia tego trybu adresowania wynika ze stosowania, np. przez kompilatory języków wysokiego poziomu, zapamiętywania danych na stosie do późniejszego przetwarzania w kolejności określonej przez regułę „pierwszy wprowadzony jest ostatnim przetwarzanym". Brak omawianego trybu adresowania wymagałby jego symulacji oznaczającej potrzebę wykonywania jałowych przesłań między rejestrami a stosem.



Rys. 6. Tryby adresowania NS16032: zewnętrzny

Tryb adresowania zewnętrznych obiektów (tryb 7, rys. 6) wykorzystuje tablicę łączącą, o której była mowa wcześniej. Przy tym trybie adresowania rozkaz zawiera indeks i wyrównanie. Indeks jest dodawany do adresu początkowego tablicy łączącej modułu, a wynik jest używany jako adres argumentu. W konsekwencji, w rozkazach programu nie są wymagane adresy bezwzględne, gdyż są zawarte w tablicy i z niej są pobierane. Odwołania do adresów w obrębie modułu są realizowane przez względne przemieszczenia liczone od początku tablicy łączącej.



Rys. 7. Tryby adresowania NS16032: indeksowy skalowany

Skalowany tryb indeksowy (tryb 8, rys. 7) jest szczególnie przydatny przy przetwarzaniu list i tablic. Użycie tego trybu adresowania pociąga za sobą wykorzystanie dodatkowego bajtu informacji określającego sposób wyznaczania argumentu. Informacja ta określa rejestr, dostarczający indeksu tablicy, i tryb adresowania, który będzie użyty do czytania adresu początkowego tablicy z pamięci. Oznacza to, że przy odwołaniu do tego trybu rozkaz używa dla jednego argumentu dwóch trybów adresowania. Pierwszy z nich służy do odczytania z pamięci adresu początkowego tablicy, a drugi — do zrealizowania dostępu do jej elementu (bajt, słowo, słowo podwójne lub poczwórne). Drugi tryb adresowania nie może być trybem indeksowym skalowanym.

W podsumowaniu warto wymienić niektóre właściwości omawianych trybów adresowania:

- uogólnienie znanych wcześniej trybów
- oszczędność kodu wynikowego generowanego przez kompilator języka wysokiego poziomu
- przystosowanie do operowania na złożonych strukturach danych
- rozszerzenie sposobów adresowania pośredniego przez pamięć
- przystosowanie do działania programu w środowisku wielozadaniowego systemu operacyjnego, z dzielonymi strukturami danych
- umożliwienie rezygnacji z programu łączącego
- skomasowanie w pojedynczych trybach adresowania funkcji użytkowych, rozdzielonych w innych realizacjach między różne tryby.

Wymienione cechy wskazują na nowatorstwo przyjętych rozwiązań. Wydaje się jednak, że efektywne wykorzystanie tych możliwości jest niełatwym zadaniem dla programisty stosującego język asemblera. Potencjalna efektywność kodu tworzonego przez kompilator, jak i spodziewane trudności w skutecznym opanowaniu języka asemblera — skłaniają do stwierdzenia o szczególnej predyspozycji mikroprocesora do programowania go w języku wysokiego poziomu.

LISTA ROZKAZÓW

Rozkazy mikroprocesorów NS16000 są kodowane na wielu (od 1 do 25) bajtach. Część adresowa pojedynczego argumentu jest zawarta w 1, 2 lub 4 bajtach pamięci. Niezwykłą cechą listy rozkazów NS16032 jest kodowanie — w oddzielnym bajcie — długości części adresowej przypisanej rozkazowi. Długość części określającej rozkaz jest również zmienna. Początki kodu rozkazu i danych będących wielokrotnością słowa nie muszą być umieszczane pod parzystym adresem — jak to jest np. dla LSI-11. Jednak dane zaczynające się od parzystego adresu są osiągane szybciej. Takie też rozmieszczenie 16-, 32- i 64-bitowych danych zaleca producent.

Rozkazy przesyłania danych

W tej grupie rozkazów wyróżnia się następujące operacje:

— przesłania dwuargumentowe
— przechowanie i odtworzenie zawartości rejestrów
— załadowanie i zapamiętanie zawartości rejestru
— przesłania adresu efektywnego.

W przesłaniach dwuargumentowych używa się pełnego zestawu trybów adresowania, wykorzystując rejestry uniwersalne tak do adresowania, jak i do pobierania oraz przesyłania argumentów. W konsekwencji stosowania odpowiednich trybów adresowania do obu argumentów możliwe są bezpośrednie transfery między komórkami pamięci bez pośrednictwa rejestrów. W tej klasie wyodrębniono specjalny rozkaz zapisywania krótkiego, 4-bitowego argumentu prostego ze znakiem.

Rozkazy zachowania i odtworzenia zawartości rejestrów dokonują transferu całych grup rejestrów na stos i z powrotem. Kolejność przesyłania — zdeterminowana wewnętrzną numeracją rejestrów, a nie kolejnością wymienienia w rozkazie asemblerowym — zapewnia ochronę przed pomyłkami przy stosowaniu dopełniających się par rozkazów „przechowaj—odtwórz".

Rozkazy załadowania i zapamiętania zawartości rejestru są stosowane dla rejestrów o funkcjach specjalnych (SP, SB, MOD, INT BASE, PSR). Niektóre z tych przesłań są zastrzeżone dla trybu pracy uprzywilejowanej.

Rozkaz transferu adresu wpisuje wygenerowany adres efektywny jednego z argumentów do drugiego.

Rozkazy arytmetyczne

Mikroprocesor NS16032 może dodawać, odejmować, mnożyć i dzielić, a oprócz tego rozszerzać znak bajtu lub słowa. Wymienione tu rozkazy występują w wariantach działających na bajtach, słowach 16- i 32-bitowych, a długość argumentu jest określana kodem rozkazu. Mikroprocesor dysponuje też rozkazami dodawania z przeniesieniem i odejmowania z pożyczką, które stosuje się do konstruowania arytmetyki wielokrotnej precyzji. Ponieważ działania dodawania i odejmowania — tak jak i inne wymienione tu rozkazy — mogą używać argumentów 8-, 16- i 32-bitowych, rozkazy z uwzględnieniem przeniesienia stosuje się przy danych o długości będącej wielokrotnością 32 bitów. Mikroprocesor dysponuje rozkazem „skróconym", którego używa dla dodawania i odejmowania 4-bitowego argumentu prostego (podobnie jak w przypadku przesłań).

Rozkazy mnożenia występują w dwóch wariantach, a oprócz tego mogą działać na trzech typach danych (bajt, słowo, długie słowo). Tak więc, rozkaz MUL (wariant 1) daje wynik tej samej długości co argumenty, przekazując tylko mniej znaczącą połowę wyniku. Rozkaz MEI (wariant 2) daje pełny wynik o długości podwójnego argumentu. Rozkaz MUL tworzy wynik ze znakiem w przeciwieństwie do rozkazu MEI, który znak ignoruje.

Rozkazy dzielenia (DIV, DIVZ i DEI) są nieco bardziej skomplikowane. Rozkaz DIV dzieli bajty, słowa i podwójne słowa i tworzy resztę tego samego rozmiaru co oba argumenty. Iloraz jednak jest zaokrąglany do największej liczby ujemnej. Tak więc, jeśli liczbę —125 podzielimy przez 60 rozkazem DIV, otrzymamy w wyniku —3 (nie —2, jak to zrobiłby rozkaz dzielenia w większości realizacji komputerowych). Z kolei rozkaz DIVZ zaokrągla iloraz do zera i w naszym przykładzie daje oczekiwany przez nas wynik —2. W obu tych dzieleniach reszta może być odzyskana przez użycie specjalnych rozkazów REM lub REMZ odpowiednio użytych po DIV i DIVZ. Trzeci rozkaz DEI jest używany do dzielnej „podwójnej" (16, 32 lub 64 bity) i dzielnika „pojedynczego" (odpowiednio 8, 16 lub 32 bity). Reszta jest po wykonaniu tego rozkazu dostępna bezpośrednio w górnej połówce dzielnika. Próba podzielenia przez zero generuje przerwanie wewnętrzne. Warto jednak zaznaczyć, że rozkazy dzielenia nie zmieniają słowa stanu PSR.
Do omawianej grupy rozkazów należą także rozkazy negacji arytmetycznej, rozszerzenia znaku i zera oraz wyznaczania wartości bezwzględnej.

Rozkazy działań logicznych

Mikroprocesor NS16032 może oczywiście wykonywać standardowe operacje logiczne sumy, iloczynu i różnicy symetrycznej oraz porównania, uzupełniania, przesunięcia cykliczne i zwykłe, a także operacje bitowe. Wykonuje je na bajtach, słowach 16- i 32-bitowych.

Rozkazy sumy, iloczynu i różnicy symetrycznej nie wpływają na stan rejestru PSR. Odpowiednie bity rejestru stanu są zmieniane przez rozkazy porównania, nie wpływające z kolei na wartości argumentów.

Rozkaz kasowania wykonuje operację iloczynu logicznego z maską, która jest negowana logicznie. Rozkaz uzupełnienia jedynkowego (negacja logiczna) jest o tyle osobliwy, że używa dwóch argumentów. Pierwszy z nich jest nie zmieniany, gdy jego adres efektywny jest różny od adresu wyniku.

Rozkazy przesunięć są dwuargumentowe. Jeden z argumentów określa licznik cykli. Adres efektywny tego licznika może być określony w dowolnym trybie adresowania. Z kolei argument poddawany obrotowi może być określony jako element znanej triady wariantów długości (bajt, słowo, długie słowo). Zauważmy, że parametryzowanie przesunięć przez licznik i długość operacji powoduje, że niektóre obroty sprowadzają się do wymiany zawartości 16-bitowych połówek argumentów (np. rejestrów) lub poszczególnych bajtów dłuższego argumentu. Przedstawiony zbiór właściwości powoduje, że rozkazy przesunięć i obrotów NS16032 tworzą najsilniejszy zbiór operacji tego typu dla mikroprocesorów 16-bitowych.

Rozkazy skoku

Rozkaz skoku bezwarunkowego (JUMP) może używać dowolnego trybu adresowania. Właściwość ta wyznacza użyteczność tego rozkazu i umożliwia np. łatwe budowanie tablic skoków.

Rozkazy skoków warunkowych występują w 14 wariantach badanych warunków i wykorzystują adresowanie względem licznika rozkazów PC. Przemieszczenie zawarte w tych skokach jest zmienne i może być zapisane na 1, 2 lub 3 bajtach pamięci. Wynika z tego, że skoki warunkowe pozwalają osiągać dowolne miejsce pamięci i optymalizować jednocześnie długość rozkazu przez wybór przemieszczenia o stosownej długości. Cecha ta odwołująca się do opisanej wcześniej właściwości listy rozkazów NS16032, które określają długość swego argumentu — nie występuje w innych mikroprocesorach. Jest atrakcyjna o tyle, że pozwala optymalizować długość kodu, szczególnie przy tworzeniu go przez kompilator języka wysokiego poziomu.
Skok z wyborem (CASE) dodaje argument do bieżącej zawartości PC. Argument jest określany przez dowolny tryb adresowania. Pozwala to organizować skoki sterowane stanem rejestru lub pamięci.

Rozkaz „dodaj, porównaj i skocz warunkowo" — znany z wcześniej omawianych mikroprocesorów — jest ogólniejszy o tyle, że argument przyrostu może być określony 4-bitową stałą, a sam argument — w dowolnym trybie adresowania. Warunek jest zdefiniowany jako badanie stanu zera wyspecjalizowanego argumentu.

Rozkazy wywołania podprogramu mogą używać adresów takiego typu jak rozkaz JUMP (dowolny tryb adresowania) lub skok warunkowy (względny adres z wyrównaniem o zmiennej długości). Wywołanie procedury zawsze powoduje przechowanie na stosie 24-bitowego adresu powrotu. Działanie procedury kończy się rozkazem powrotu z podprogramu (RET), powrotu z przerwania (RETI) lub powrotu z przerwania wewnętrznego (RETT). Interesującą właściwością rozkazów RET i RETT jest istnienie argumentu będącego stałą. Stała ta określa, ile komórek pamięci kodu rozkazów ma być przeskoczonych przy powrocie, w stosunku do adresu przekazanego na stos w wyniku wywołania lub przerwania wewnętrznego. Właściwość ta pozwala przekazywać parametry wywołania w komórkach następujących po rozkazie skoku do procedury. Metoda ta — nagminnie stosowana np. przy komunikacji z systemem operacyjnym przez wywołania przerwań wewnętrznych lub przy przekazywaniu parametrów wołanych przez adres (FORTRAN) — upraszcza obsługę powrotu do programu głównego przez brak potrzeby modyfikowania szczytowego elementu stosu. Mikroprocesor NS16032 ma też specjalne rozkazy wywołania procedur zewnętrznych i odpowiednie rozkazy dla powrotu z takiej procedury. Wywołania te, odwołując się do procedury przez tablicę łączącą, przechowują dodatkowo (odtwarzany przy powrocie) stan rejestru MOD.

Rozkazy operowania napisami i blokami danych

Podobnie jak I-8086 mikroprocesor NS16032 dysponuje trzema prostymi rozkazami operowania napisami. Rozkazy z tej grupy, a także opisywane dalej przesłania blokowe przypisują funkcje rejestrom, które w kontekście tych rozkazów przestają być uniwersalne i symetryczne względem zastosowań. Tak więc rozkazy z tych grup angażują rejestry R0,..., R4, przypisując im następujące znaczenia:

R0 — licznik
R1 — adres początkowy ciągu 1
R2 — adres początkowy ciągu 2
R3 — adres początkowy tablicy tłumaczącej
R4 — wyróżniony znak kończący ciąg.

Rozkazy operacji napisowych mogą porównywać elementy, przesyłać je i przeskakiwać. Elementami ciągu mogą być bajty, słowa 16- lub 32-bitowe. Adresy z rejestrów R1 i R2 mogą przyrastać lub zmniejszać się. Jednocześnie, w trakcie przeglądania ciągu jego elementy mogą być przekształcane wg wzorca zapisanego w tablicy wskazywanej przez R3 (tłumaczenie). Warunkiem końca operacji jest wyzerowanie licznika R0, osiągnięcie wzorca z R4 lub rozpoznanie elementu różnego od przekazanego wzorca. Rozkazy z tej grupy mogą być przerywane przez zgłoszenia zewnętrzne po zakończeniu kroku związanego z modyfikacją rejestrów adresowych.

Mikroprocesor NS16032 ma dwa rozkazy z grupy przesłań blokowych. Pierwszy z nich (MOVM) przesyła blok informacji, drugi (CMPM) — porównuje dwa bloki danych.

Obydwa te rozkazy działają na blokach o długości określonej i ograniczonej z góry do 16 słów.

Rozkazy operacji bitowych

Rozkazy z tej grupy mogą ustawiać, kasować i testować bity. Wynik działania odpowiedniego rozkazu jest pojedynczym bitem zapamiętywanym w rejestrze lub pamięci. Używają dowolnego trybu adresowania do określenia argumentów oznaczających: przeznaczenie (pamięć lub rejestr) i położenie bitu w słowie podlegającym operacji. Położenie to (argument efektywny) może przyjmować wartości z zakresu od 0 do 31. Wśród rozkazów bitowych istnieją także rozkazy wyszukiwania pierwszego bitu ustawionego na 1 oraz rozkaz inwersji bitu. Rozkaz inwersji bitu występuje w wariancie wykonującym zaprzeczenie stanu najmniej znaczącego bitu argumentu źródłowego z przepisaniem do argumentu wynikowego.

Do rozkazów bitowych zalicza się także operacje ustawiania bitu wskazanego argumentem wg wartości wybranego bitu rejestru stanu PSR.

Rozkazy operacji na polach bitowych

W wielu zastosowaniach wymagających upakowania danych pojawia się problem efektywnego adresowania obiektów, które nie dają się wpasować w bajtową strukturę pamięci. Mikroprocesor NS16032 dysponuje rozkazami rozwiązującymi ten problem. Używają one trzech agrumentów, z których dwa są określane przez normalne tryby adresowania, a trzeci jest stałą opisującą długość pola bitowego. Rozkazy z tej grupy pozwalają ustawiać bity wskazanego pola i odczytywać je.

Rozkazy rezerwacji i obsługi tablic

Rozkazy z tej grupy pozwalają obliczać indeksy wielowymiarowych tablic i sprawdzać, czy wartość wyliczonego indeksu nie przekracza założonego zakresu. Wynik testu jest przekazywany w bicie F rejestru stanu PSR i może być badany rozkazem warunkowego przerwania wewnętrznego.

Mikroprocesor dysponuje też rozkazami przydatnymi przy wywołaniu procedur. Te rozkazy to ENTER i EXIT. Można je zaliczyć — tak jak wyznaczanie i sprawdzanie indeksu — do rozkazów ułatwiających gospodarkę pamięcią danych. Pierwszy z nich (ENTER) wykonuje sekwencje działań poprzedzającą wejście do procedury ze zmiennymi lokalnymi i służy do zarezerwowania na stosie pola na te zmienne. Sekwencja działania rozkazu ENTER realizowana jest w trzech krokach:

— przepisanie rejestru ramki FP na stos
— przepisanie wskaźnika stosu SP do FP
— odjęcie stałej od SP.

Wykonanie rozkazu ENTER powoduje więc zarezerwowanie na stosie obszaru, którego adres zawiera rejestr FP, a jego długość jest określona wartością stałej odejmowanej w kroku 2. Pole to może być używane na zmienne lokalne procedury, która może być uaktywniona po wykonaniu rozkazu ENTER. Rekurencyjność tak wywołanej procedury zapewnia przechowanie na stosie starej wartości FP. Rozkaz EXIT wykonuje działania odwrotne do rozkazu ENTER, zwalniając pole zarezerwowane i odtwarzając stan wskaźników SP i FP.

Rozkazy specjalne

Do tej grupy rozkazów można zaliczyć rozkaz oczekiwania (WAIT), rozkaz pusty (NOP) oraz rozkazy przerwań wewnętrznych. Rozkaz WAIT przestawia procesor w stan oczekiwania, z którego może być wyprowadzony przez przerwanie. Rozkaz NOP pozwala wytracić kwant czasu bez wpływu na stan procesora.

Rozkazy przerwań wewnętrznych — to dwa rozkazy bezwarunkowych przerwań zaprojektowane z intencją komunikowania się z systemem operacyjnym (SVC, ang. supervisor call) i wstawieniem przerwania w program uruchamiany (BPT, ang. breakpoint trap). Trzecie przerwanie wewnętrzne — FLAG — jest uaktywniane warunkowo, gdy bit F rejestru stanu PSR jest ustawiony. Bit ten jest modyfikowany przez wiele rozkazów, np. w wyniku odpowiedzi na badanie dozwolonego zakresu indeksów. Może być więc wykorzystywany do reagowania na sytuacje rozpoznane jako poważne, wymagające systemowej obsługi błędy programu.

Do rozkazów specjalnych można też zaliczyć rozkaz SETCFG (ang. set configuration). Może on ustawiać bity C, M, F i I, które są tylko zapisywalnymi bitami rejestru stanu PSR. Bity te określają konfigurację w tym sensie, że ich stan odzwierciedla np. obecność podległych mikroprocesorów systemu mikrokomputerowego (np. procesor operacji zmiennoprzecinkowych, zarządzania pamięcią, sterowniki urządzeń itp.).

Rozszerzanie listy rozkazów

Lista rozkazów NS16032 może być rozszerzona w miarę pojawiania się w konfiguracji mikrokomputera nowych układów (mikroprocesorów) podległych. Takimi podległymi układami mogą być procesory zmiennoprzecinkowe, układy zarządzania pamięcią itd.

Gdy NS16032 rozpozna kod niewykonywalnego rozkazu próbuje przekazać jego wykonywanie do procesora podległego, gdy taki jest w zestawie. Gdy i ta próba nie powiedzie się — generowane jset przerwanie wewntęrzne. Oczywiste jest, że brak procesora podległego może być rozpoznany wcześniej wg stanu zapisanego do PSR przez rozkaz SETCFG.

Konkurs na grę edukacyjną

Warszawski Klub Mikrokomputerowy ABAKUS proponuje wszystkim zainteresowanym współudział w tworzeniu programów edukacyjnych.

Ogłasza konkurs na pomysł gry edukacyjnej oraz na program takiej gry.

Gra może być przeznaczona dla osób w dowolnym wieku, a i sam materiał nauczający może być dowolny. Bardzo pożądane są programy uniwersalne, tzn. mogące znaleźć zastosowanie w różnych dziedzinach wiedzy.

Konkurs ma charakter powszechny. Praca konkursowa na pomysł powinna być przedstawiona w formie kilkustronicowego opisu (maksimum 600 słów) — do 15 listopada br. Natomiast program powinien zostać zrealizowany (do 30 marca 1984) na jednym z mikrokomputerów dostępnych w Klubie: ZX81 (do 64 K), SPECTRUM (do 48 K), ORIC-1 (do 64 K); i przysłany w formie zapisu na kasecie magnetofonowej. Jeśli program ma być zaprezentowany na innym sprzęcie, uczestnik będzie zmuszony do przedstawienia go w Klubie razem z mikrokomputerem, po uprzednim listownym uzgodnieniu spotkania. Wymagane jest oświadczenie autora o oryginalności programu.

Prace oceniać będzie specjalna komisja powołana przez Radę Klubu — w skali od 0 do 100. Punkty będą przyznawane następująco:

- do 40 — za element gry i zabawy
- do 30 — za szybkość i jakość nauczania
- do 15 — za uniwersalność
- do 5 — za walory wizualno-dźwiękowe
- do 5 — za dostępność mikrokomputera
- do 5 — za wykorzystanie możliwości mikrokomputera.

Prace należy nadsyłać pod adresem: **Klub Mikrokomputerowy „ABAKUS" ul. Sobieskiego 18. 02-930 Warszawa.** Konsultacje mają być dokonywane także bezpośrednio w Klubie, podczas otwartych spotkań w poniedziałki w godz. 17—20. Istnieje możliwość korzystania ze sprzętu klubowego.

Pierwszą nagrodą w konkursie na pomysł jest mikrokomputer ZX81 (1 K), a w konkursie na program ZX81 (16 K). Fundatorem nagród jest firma polonijna AMEPROD z Poznania. Ponadto wszystkie osoby proponujące ciekawe prace mogą zostać członkami Klubu.

JERZY KARCZMARCZUK
Instytut Fizyki
Uniwersytet Jagielloński
Kraków

Jeszcze o ZX81

Poniższy artykuł jest częściowo polemicznym uzupełnieniem artykułu Jana Dawidowskiego („Komputer osobisty ZX81") z nr 5/1983 INFORMATYKI. Jest to opis mikrokomputera ZX81 oparty o ponad roczne doświadczenia jego użytkownika; opis, który może być przydatny zarówno osobom programującym w BASICU, jak i zainteresowanym możliwą rozbudową systemu.

ZASILANIE I OBSŁUGA URZĄDZEŃ ZEWNĘTRZNYCH

ZX81 jest zasilany prądem stałym o wewnętrznie stabilizowanym napięciu 5 V. Standardowy zasilacz SINCLAIRA dostarcza prądu o napięciu 9 V i maksymalnym poborze prądu 1200 mA, co dwukrotnie przekracza potrzeby samego komputera i umożliwia jednoczesne zasilanie dodatkowych urządzeń, np. drukarki czy dodatkowej pamięci. Przy dramatycznych spadkach napięcia w sieci, co niestety u nas czasami występuje, konieczne jest stosowanie stabilizatora napięcia sieciowego, zwłaszcza jeżeli komputer współpracuje z zewnętrznym blokiem pamięci.

Końcówki kabli połączeniowych sprzęgających ZX81 z telewizorem i magnetofonem nie są — jak twierdzi Dawidowski — czymś niedostosowanym do polskiego sprzętu. Telewizor łączy się za pomocą kabla koncentrycznego i wtyku koncentrycznego, który jest standardowy dla fal decymetrowych (UHF). Jest to jedyne wejście antenowe np. w telewizorach „Neptun".

Pewien kłopot występuje w przypadku starszych telewizorów. Częstotliwość generowanej fali nośnej odpowiada kanałowi 36, zatem w przypadku posiadania odbiornika nie dostosowanego do odbioru fal decymetrowych wymagana jest przeróbka modulatora znajdującego się w komputerze. Taką przeróbkę powinien przeprowadzić każdy wykwalifikowany elektronik.

Końcówki magnetofonowe są typu „jack" 3,5 mm. Większość polskich magnetofonów ma natomiast wejście typu DIN, jest to więc rzeczywiście pewien problem. Niemniej w handlu znajdują się obecnie części umożliwiające tę banalną przeróbkę.

Znacznie bardziej kłopotliwy niż brak wyłącznika zasilania jest brak przycisku „Reset", który umożliwiałby powtórne zainicjowanie pracy procesora bez wyłączania zasilania. Problem polega na tym, że w przypadku zawieszenia się systemu — co może mieć miejsce na skutek błędów w programie, próby wczytania z kasety źle nagranego programu lub wyładowania elektryczności statycznej (uwaga na dywany!) — jedynym lekarstwem jest wyłączenie komputera. Można temu zaradzić instalując własny przycisk inicjowania; może to być zwykły przycisk dzwonkowy podłączony do szyny zewnętrznej między linią RESET procesora a ziemią.

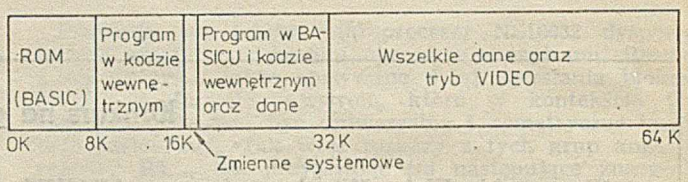
Zewnętrzna szyna mikroprocesora jest wyprowadzona na płytkę montażową i dostępna przez szczelinę w obudowie komputera, np. przez złącze krawędziowe produkowane w Polsce przez ELTRĘ (typ 801).

Podstawowym wejściem ZX81 jest miniaturowa klawiatura membranowa (nie — sensorowa) o 40 płaskich, dotykowych przyciskach. Przyciski są wielofunkcyjne i dostosowane do języka BASIC. Zwiększa to nieco szybkość pisania programów, ale i tak trudno tę klawiaturę nazwać wygodną. Wymaga ona wielogodzinnego przyzwyczajania się, ze względu na jej niewielki rozmiar i brak mechanicznej reakcji ze strony przycisków. W związku z tym dostępne na rynkach zachodnich proste i tanie (ale większe) mechaniczne klawiatury do ZX81 — robią furorę. Procesor odczytuje znaki z klawiatury czysto programowo, bez żadnych przerwań, traktując ją jako zbiór ośmiu pięciobitowych portów wejściowych. W przypadku zawieszenia systemu klawiatura staje się nieczynna.

Moim zdaniem — standardowa klawiatura bardzo obniża, jeśli nie dyskwalifikuje możliwości zastosowania komputera ZX81 do celów dydaktycznych. W komputerze SPECTRUM, który jest następnym komputerem firmy SINCLAIR (a nie — jak twierdzi Dawidowski — wersją rozwojową ZX81) zastosowano już inną klawiaturę.

STRUKTURA PAMIĘCI KOMPUTERA

Schemat pamięci komputera ZX81 w wersji maksymalnej przedstawiono na rysunku. Z pełnego obszaru adresowego procesora Z80 (64 K = 65536) dolne 8 K bajtów jest zajęte przez pamięć ROM, zawierającą: interpreter BASICA, procedury obsługi ekranu i klawiatury, itp. Reszta jest w zasadzie do dyspozycji użytkownika; należy jednak pamiętać, że zarówno sprzęt, a dokładniej sterownik systemu, jak i niektóre procedury zawarte w pamięci ROM wykorzystują pozostałą pamięć selektywnie i swoboda użytkownika jest znacznie ograniczona.



Schemat pamięci mikrokomputera ZX81 w wersji maksymalnej (bez rozszerzeń sprzętowych)

Bajty od 16384 do 16508 zawierają tzw. zmienne systemowe: tablice adresów, liczniki itp. Naruszenie integralności niektórych z nich może uniemożliwić współpracę z BASICIEM, a nawet zawiesić system. Przez dłuższy czas maksymalnym rozszerzeniem pamięci ZX81 było 16 K bajtów adresowane od 16 do 32 K. Takie właśnie maksymalne rozszerzenie przewidzieli konstruktorzy komputera i jego oprogramowania. Aktualnie są dostępne rozszerzenia do pełnych 64 K bajtów, tj. z 56 K pamięci operacyjnej. Trzeba jednak mieć na uwadze następujące ograniczenia:

- Interpreter BASICA „nie widzi" obszaru pamięci pomiędzy 8 a 16 K. Ten obszar jest dostępny dla programów w kodzie maszynowym i ich danych. BASIC może wykorzystywać ten obszar tylko przez instrukcje PEEK, POKE i USR.

● Program w BASICU nie może być za długi, musi mieścić się w obszarze do 32 K, w przeciwnym wypadku procedura obsługi ekranu TV może się zawiesić (z przyczyn opisanych poniżej). Dane dla programu mogą zajmować dowolny obszar pamięci — aż do 64 K.

● Program w kodzie maszynowym musi być również umieszczony w dolnej połowie obszaru adresowego, tj. do 32 K. Program może korzystać z pola 32 K—64 K jako z danych, ale żadnej instrukcji znajdującej się w tym polu mikroprocesor nie może wykonać. Próba skoku do instrukcji o adresie większym niż 32 K jest traktowana specjalnie przez sterownik systemu, zwany w dalszym ciągu artykułu SCL (Sinclair Computer Logic), na temat którego firma SINCLAIR odmawia wyjaśnień.

OBSŁUGA EKRANU TV

ZX81 nie generuje sygnału fonii. Elementy wyświetlanego obrazu są białe lub czarne, bez pośrednich stopni szarości. ZX81 nie wykorzystuje pełnego ekranu, a jedynie jego centralny prostokątny fragment obejmujący ok. 87% powierzchni ekranu i zawierający 192 linie poziome, z których każda zawiera 256 elementów obrazu.

Użytkownik ma bardzo ograniczone możliwości komponowania obrazu. Nie można zaprogramować dowolnie wyświetlanych elementów; można wykorzystać jedynie stały, wbudowany zestaw znaków alfanumerycznych i graficznych, liczący 64 znaki pozytywowe (czarne znaki na białym tle) i 64 znaki negatywowe. Każdy znak zajmuje na ekranie kwadrat 8×8 elementów obrazu — tak, że możliwe jest wyświetlenie 24 wierszy po 32 znaki w wierszu. Nie można zatem rysować wykresów na ekranie, gdyż możliwości graficzne urządzenia są zbyt prymitywne.

Szereg firm oferuje dodatkowy sprzęt umożliwiający otrzymanie tzw. grafiki o dużej zdolności rozdzielczej, tj. sterowanie każdym elementem obrazu z osobna. Uzyskanie takiej możliwości (programowo) być może nie jest niemożliwe, ale wydaje się bardzo trudne. Powody tego pesymizmu omówiono poniżej, przy wyjaśnianiu technicznych szczegółów generowania obrazu. Szczegóły te są bardzo istotne z punktu widzenia możliwości rozbudowy systemu.

Każdy znak wysyłany na ekran zajmuje jeden bajt w ruchomej tablicy, zwanej dalej DF (ang. Display File). Tablica ta jest podzielona znakami zmiany wiersza, których kod — 76_{16} — nieprzypadkowo pokrywa się z kodem instrukcji HALT mikroprocesora Z80. Każdy kod znaku ma swój odpowiednik w pamięci ROM — 8-bajtową tablicę, której bity odpowiadają elementom obrazu tworzącym znak na ekranie. Tablica ta będzie nazywana generatorem znaku. Wyświetlenie znaku polega więc na pobraniu kodu znaku z DF i — następnie — na ośmiokrotnym odwołaniu się do kolejnych bajtów generatora w ośmiu kolejnych liniach wysyłanych na ekran. Procesor inicjuje przesyłanie znaków przez pobranie adresu DF, dodanie do tego adresu liczby 2^{15}, tj. ustawienie najbardziej znaczącego bitu, a następnie wykonanie skoku do adresu wynikowego. Taka sytuacja jest rozpoznawana przez SCL, który przejmuje kontrolę nad szyną danych i częściowo nad szyną adresową. Wynika stąd niemożność wykonania programu w kodzie maszynowym, umieszczonego w górnej połowie obszaru adresowego. Stąd też wynika warunek (jednoznaczności), że DF musi znajdować się w dolnej połowie obszaru adresowego. Ponieważ BASIC umieszcza DF zaraz po końcu tekstu programu, powyższy warunek ogranicza długość programu w BASICU.

Po wykonaniu wspomnianego skoku procesor przechodzi w tryb pracy przypominający pracę sterownika DMA. Przez swój licznik rozkazów adresuje kolejne słowa DF; zaś SCL, który po rozpoznaniu bitu 15 zaraz go zeruje, nie dopuszcza do procesora wysyłanych z pamięci kodów znaków, generując w ich miejsce rozkaz pusty NOP (No Operation). Zatem Z80 — poza generowaniem kolejnych adresów — nie robi nic, a właściwie prawie nic. Należy bowiem wiedzieć, że unikalną cechą mikroprocesora Z80 jest generowanie tzw. adresów odświeżania pamięci dynamicznych. Generowanie to odbywa się w tzw. cyklu odświeżania, który obejmuje dwa cykle zegara, w czasie gdy mikroprocesor zajmuje się wyłącznie wewnętrznymi przesłaniami związanymi z rozkodowaniem rozkazu, i szyna adresowa jest wolna.

ZX81 wykorzystuje cykl odświeżania do adresowania generatora znaków. Adres generatora jest tworzony przez SCL przy wykorzystaniu tzw. rejestru przerwań procesora, który zawiera adres początku tablicy generatorów. Natomiast rejestr odświeżania, którego zawartość jest zwiększana o 1 po każdym cyklu procesora, służy jako licznik znaków w wierszu. Po przesłaniu jednej linii na ekran, SCL generuje przerwanie, a procedura obsługi tego przerwania sprawdza, czy wiersz znaków został już osiem razy przejrzany i — odpowiednio ustawia adresy.

Jednoczesna praca obliczeniowa i obsługa ekranu nie są więc możliwe. W opisanym trybie pracy, podczas obliczeń, ekran jest szary. Wyświetlenie następuje wówczas, gdy procesor, nie robiąc nic innego, czeka na rozkaz z klawiatury, która jest czytana podczas okresu wygaszenia pionowego. Jest to bardzo niewygodne, jeżeli wyświetlenie obrazu jest istotą pracy komputera, a więc przy przeglądaniu zmieniających się danych, przy różnych grach komputerowych itp.

ZX81 ma jednak możliwość pracy w innym trybie — zwanym SLOW (powolny), w odróżnieniu od omówionego wyżej trybu FAST. W trybie SLOW co 1/50 sekundy SCL generuje niemaskowalne przerwania i oddaje sterowanie procedurze obsługi ekranu, która zwraca sterowanie programowi po wyświetleniu pełnego ekranu. W ten sposób wyświetlenie następuje w sposób ciągły, z migotaniem prawie niezauważalnym dla oka, natomiast obliczenia trwają przeszło cztery razy dłużej.

Z powyższego opisu wynikają następujące uwagi dotyczące możliwości sprzętowych rozszerzeń układu:

● Dołączanie dodatkowej półprzewodnikowej pamięci dynamicznej nie wchodzi w grę, chyba że nie będzie ona w żaden sposób związana z obsługą ekranu i będzie wyposażona we własny układ odświeżania.

● Dołączenie zewnętrznych urządzeń z wykorzystaniem mechanizmu przerwań jest bardzo kłopotliwe. Ponieważ dolne 8 K bajtów pamięci jest zajęte, ani przerwania niemaskowalne, ani maskowalne w tzw. trybie 1 nie wchodzą w rachubę, gdyż ich procedury obsługi związane są z generowaniem obrazu i nielegalne ich użycie z reguły zawiesza system. Przerwania w trybie 0 mogą co najwyżej (przez instrukcję RST) oddać sterowanie systemowi BASIC, natomiast przerwania w trybie 2 mogą odbywać się jedynie w czasie, gdy ZX81 nie generuje obrazu, gdyż ten tryb wykorzystuje rejestr przerwań zawierający standardowo adres tablicy generatorów znaków.

● Nie wszystkie adresy portów wejściowych i wyjściowych są do dyspozycji użytkownika. Niektóre z nich odpowiadają specjalnym rejestrom SCL, sterującym przerwaniami. Należy ponadto pamiętać, że klawiatura, magnetofon oraz drukarka mają również przypisane stałe adresy portów.

Tak więc, wbrew temu co twierdzi Jan Dawidowski, łatwość rozbudowy konfiguracji nie jest najmocniejszą stroną ZX81 i wymaga uwzględnienia wielu czynników ograniczających. Nie jest to jednak niemożliwe, np. stosunkowo łatwo można sprząc ZX81 z jakimś wolnym urządzeniem wyjściowym, np. drukarką mozaikową DZM.

OBSŁUGA MAGNETOFONU

Wyjściem akustycznym komputera jest jednobitowy rejestr w SCL, którego stan może być programowo zmieniony przez mikroprocesor. W ten sposób Z80 generuje prostokątny sygnał akustyczny o strukturze zadanej przez zaprogramowanie pętli o odpowiednim czasie trwania. Sygnał ten jest przepuszczany przez prosty filtr pasmowy i może służyć do nagrywania programów na kasetę magnetofonową. Standardowe procedury zapisują program z szybkością ok. 250 bodów. Jest to niewygodne — nawet stosunkowo niedługie programy zapisuje się przez kilka minut, co zwiększa możliwość przekłamań. Standardowe procedury zapisują program w kodzie trochę zbliżonym do standardu Kansas City [1]), ale jednak innym, co uniemożliwia przenoszenie programów na inne komputery. Stosun-

[1]) Logicznej jedynce i zeru odpowiadają impulsy o różnej częstotliwości, np. dla kodu Kansas City — 2400 Hz i 1200 Hz — przyp. red.

kowo łatwo można jednak napisać w kodzie własne procedury, które umożliwiają zapis z szybkością przekraczającą 4000 bodów, pod warunkiem posiadania magnetofonu i kaset dobrej jakości.

Standardowe procedury mają jedną zaletę, o której można wspomnieć: dzięki małej częstotliwości i wąskiemu widmu sygnału można to widmo przesyłać standardowymi łączami telefonicznymi (niestety, nie sprawdziłem tego w Polsce), a także nagrywać inne nośniki, np. elastyczne „pocztówki" dźwiękowe. Tę drogę wybrało francuskie czasopismo MICRO SYSTÈMES, które w tegorocznym majowym numerze zaoferowało posiadaczom ZX81 telewizyjną grę komputerową na płycie!

BASIC

Omówię tu jedynie kilka cech języka i jego realizacji charakterystycznych dla ZX81, które nie zostały należycie wyeksponowane przez Jana Dawidowskiego.

Różnice w stosunku do standardu są niewielkie. Przy przenoszeniu programów napisanych w innych wersjach BASICA należy uwzględnić brak instrukcji DATA i związanych z nią READ i RESTORE. Przetwarzanie tekstów (ciągów znaków) również odbiega nieco od standardu. Zmienne tekstowe mogą zawierać tekst dowolnej długości, natomiast tablice tekstowe — o ustalonej liczbie znaków. Interesującym rozwiązaniem składniowym jest jednolita notacja dla wycinków tekstu. Pisząc np. A$(N TO M) — z możliwością opuszczenia zarówno N, jak i M — zastępujemy cztery niedostępne w BASICU ZX81 funkcje LEFT$, RIGHT$, MID$ oraz TL$. Kody znaków nie odpowiadają standardowi ASCII, co trzeba uwzględniać przy przenoszeniu programów, które dokonują konwersji liczb na teksty lub odwrotnie. W mikrokomputerze SPECTRUM firma zastosowała już kod ASCII.

Pięciobajtowa reprezentacja liczb zmiennopozycyjnych, która umożliwia dokładność ok. 9,5 cyfr dziesiętnych i zakres ok. 10±38, dla wielu zastosowań jest wystarczająca. Cała arytmetyka jest zmiennopozycyjna, co niejednokrotnie niepotrzebnie przedłuża obliczenia. W nowszych wersjach BASICA istnieje opcja DEFINT, umożliwiająca interpreterowi BASICA deklarowanie niektórych zmiennych jako całkowitych, jednakże tutaj nie występuje. Niestety, standardowe procedury arytmetyczne zawierają pewne błędy, które powodują, że błędy zaokrąglenia narastają w trakcie obliczeń szybciej niż powinny. Do zauważonych cech tego rodzaju należą np.:

● $2^{-32}+1-1 = 2^{-31}$, co oznacza, że nie zawsze można mieć zaufanie do przedostatniego bitu

● 0,00001 różni się od 1/100000 o 2^{-48}, co oznacza, że procedura czytająca liczbę dziesiętną i przetwarzająca ją na postać dwójkową traktuje bity zaokrąglenia inaczej niż procedura dzielenia.

Z kolei program rozwiązujący układ równań liniowych prostą metodą Gaussa (szczegóły w redakcji), przetestowany dla macierzy Hilberta o wymiarach 4×4, wykazał błąd względny ok. $3{,}5 \cdot 10^{-7}$. Przy zwiększaniu wymiaru układu równań błąd rośnie wręcz dramatycznie (nawet biorąc pod uwagę słabe uwarunkowanie tego układu równań), co powoduje, że obliczenia na ZX81, które wiążą się z wielokrotnymi iteracjami, np. rozwiązywanie równań różniczkowych czy układów równań liniowych wysokich rzędów, wymagają bardzo stabilnych metod numerycznych.

Poniżej podaję kilka wyników szacunkowych pomiarów czasu operacji arytmetycznych. Pomiar został przeprowadzony „in vivo" przy użyciu programu zawierającego pętlę i stopera; oprócz czasu trwania samej operacji obejmuje również czas, w którym interpreter rozkodowywuje znak operacji i przekazuje sterowanie odpowiedniej procedurze:
— dodawanie i odejmowanie: od 1,3 do 2 ms — w zależności od różnicy rzędów obu liczb, tj. od czasu potrzebnego na skalowanie
— mnożenie: ok. 2, 3 ms
— dzielenie: ok. 2, 9 ms.

Podane czasy dotyczą operacji na stałych. Operacje na zmiennych trwają znacznie dłużej, ponieważ interpreter nie pamięta adresów zmiennych, które są ruchome; przy każdym odniesieniu do zmiennej przeszukuje tablicę zmiennych, aż znajdzie odpowiednią nazwę.

Standardowe funkcje arytmetyczne wykorzystują rozwinięcia na wielomiany Czebyszewa i są dość wolne:

EXP — 0.04 s

LN — 0.07 s

SQR — 0.11 s

SIN, COS — 0.04 s

TAN — 0.08 s

ATN — 0.06 s

ASN, ACS — 0.18 s

Wyniki będą nieco inne, jeśli np. argumentem funkcji SIN będzie bardzo duża liczba, którą interpreter musi najpierw sprowadzić do przedziału $0 - \pi/2$. Potęgowanie zabiera ok. 0,11 s. Na uwagę zasługuje bardzo długi czas liczenia pierwiastka kwadratowego. Jest on liczony jako eksponenta z 1/2 razy logarytm argumentu, co jest bardzo nieefektywne. Podobnie — funkcje arcsin i arccos są liczone za pomocą funkcji arcus tangens i pierwiastka. Projektanci BASICA ZX81 ewidentnie poświęcili szybkość obliczeń na rzecz zwartości kodu.

Na uwagę zasługuje konstrukcja bloku sterowania procedurami arytmetycznymi interpretera. Został on zaprojektowany jako „kalkulator wirtualny", tj. jako maszyna ze stosem, posiadająca swój własny język wewnętrzny o jednobajtowych kodach operacji. Ten kalkulator można wykorzystać z wnętrza programu w kodzie wewnętrznym i napisać własne, szybkie procedury obliczania funkcji elementarnych czy specjalnych, a także realizować bardziej złożone operacje arytmetyczno-logiczne z pominięciem BASICA. Jest to niestety dość kłopotliwe, gdyż BASIC ZX81 nie przewiduje żadnej wygodnej metody przekazywania parametrów podprogramom, zarówno w kodzie wewnętrznym, jak i w BASICU.

Absolutnie nie zgadzam się ze zdaniem Jana Dawidowskiego, że BASIC jest jedną z zalet ZX81, natomiast w pełni zgadzam się ze zdaniem E. W. Dijkstry (cytowanym na łamach INFORMATYKI), który uważa BASIC za jeden z najbardziej antydydaktycznych języków programowania na świecie. Niestety projektanci oprogramowania ZX81 niewiele zrobili w kierunku poprawy języka pod kątem jego modularności. W obecnej wersji nie można definiować własnych funkcji, wszystkie podprogramy są bezparametrowe, wszystkie zmienne są globalne. Procedury wydruku i redagowania są bardzo prymitywne; pod tym względem wszystkie sprzedawane w Polsce minikomputery, jak MERA-60, NEPTUN, czy PSPD-90 są oprogramowane lepiej.

Poważną wadą ZX81 jest słabo opracowana współpraca z kasetową pamięcią zewnętrzną. Pomijając już powolność odczytu i zapisu, możliwy jest jedynie zrzut (ang. dump) pełnych programów i ich wczytanie. Nie można zapisywać ani plików z danymi, ani fragmentów programów, nakładek czy podprogramów. Załadowanie programu z kasety kompletnie kasuje poprzednią zawartość pamięci, którą dysponuje BASIC. Uniemożliwia to tworzenie i przechowywanie na taśmach bibliotek podprogramów. Oczywiście, wady te zostały w pewnym stopniu złagodzone przez niezależnych producentów oprogramowania — zarówno amatorów, jak i profesjonalistów. Oferowany jest bogaty garnitur edytorów, kompilatorów itp. Jeżeli komputery ZX81 mają znaleźć w Polsce niebanalne zastosowanie, taki rynek musi powstać i u nas.

Algorytmy „kalendarzowe"

W artykule tym zajmiemy się czterema algorytmami związanymi z kalendarzem. Będą to:
— obliczanie liczby dni dzielących dwie daty
— obliczanie daty po upływie pewnego czasu od danego dnia
— obliczanie dnia tygodnia na podstawie daty
— znajdowanie daty K-tego w danym miesiącu j-tego dnia tygodnia (np. druga środa w marcu 1957).

Założeniem naszych programów jest ich „chytrość" i stosunkowo krótki czas działania. Zapisujemy je w BASICU ze względu na ich prostotę i ewentualne zainteresowanie nimi użytkowników mikrokomputerów.

Używać będziemy standarowych funkcji INT (część całkowita), MOD (reszta z dzielenia) i SGN (znak argumentu, tzn. 1 — gdy argument jest dodatni, -1 — gdy ujemny i 0 — gdy argument równa się zero). Rozumieć będziemy przy tym, że rok R jest rokiem przestępnym, jeżeli:
R **mod** 4 = 0 i R **mod** 100 ≠ 0 lub R **mod** 400 = 0
to znaczy lata 1600, 1980, 1984 2000 są przestępne, zaś lata 1700, 1900, 1975 — zwykłe.

Obliczanie liczby dni pomiędzy dwoma datami

Napiszmy najpierw procedurę pomocniczą, która dla danej daty D, M, R (tzn. Dzień, Miesiąc, Rok) oblicza ile czasu minęło od dnia 1 stycznia 1 roku naszej ery (przy fałszywym założeniu, że kalendarz nie ulegał zmianom — nie będzie nam to jednak przeszkadzało w dalszych rozważaniach). Ma ona postać następującą:

```
100 REM DATA JEST DANA W POSTACI TRZECH LICZB D, M, R
110 REM CZAS JAKI UPŁYNĄŁ OD 1.01.1
115 REM ZWRACANY JEST NA ZMIENNEJ P
120 IO = SGN (INT(N/3))
130 I1 = 1 — IO
140 P = (R—1) * 365+(M—1) * 31+D—1
150  P = P+INT ((R—I1)/4) — INT ((R—I1)/100)
155  P = P+INT ((R—I1)/400)
160  P = P—IO * INT (M * 0.4+2.3)
170 RETURN
```

Teraz dla danych dat D1, M1, R1 i D2, M2, R2 obliczamy ilość dni jakie je dzielą:

```
200 REM WYNIK ZWRACANY JEST NA ZMIENNEJ P
210 D = D1
220 M = M1
230 R = R1
240 GOSUB 100
250 P1 = P
260 D = D2
270 M = M2
280 R = R2
290 GOSUB 100
300 P = P—P1
310 RETURN
```

Obliczanie daty po upływie pewnej liczby dni od danego dnia

Procedura zakłada, że datę danego dnia zawierają zmienne D, M, R, natomiast liczba dni, jakie upłynęły (lub upłyną) od tej daty — zmienna O. Wynikową datę otrzymamy również na zmiennych D, M, R.

Algorytm w pierwszej fazie znajduje datę przybliżoną, a następnie w sposób iteracyjny dokonuje poprawek aby ostatecznie podać datę dokładną. Przeprowadzone testy wykazały, że pętla 480—630 wykonuje się ok. 2—3 razy. Tekst procedury ma postać następującą:

```
400 GOSUB 100
410 P1 = P+O
420 I = INT (P1/365)
430 J = INT (I/4) — INT (I/100)+INT (I/400)
440 P2 = P1—J
450 R = INT (P2/365)+1
460 M = INT (MOD (P2, 365)/31)+1
470 D = 28
480 GOSUB 100
490 IF P = P1 THEN RETURN
500 IF P < P1 THEN 580
510 D = D—P+P1
520 IF D > O THEN 480
530 D = 31
540 M = M—1
550 IF M > O THEN 480
560 M = 12
565 R = R—1
570 GOTO 480
580 D = 28
590 M = M+1
600 IF M < 13 THEN 480
610 M = 1
620 R = R+1
630 GOTO 480
```

Obliczanie dnia tygodnia na podstawie daty

Przedstawioną poniżej procedurę prezentowaliśmy już (zapisaną w ALGOLU) w INFORMATYCE nr 11—12/81. Założenia procedury są następujące: na zmiennych D, M, R podana jest data, na podstawie której obliczamy przypadający na nią dzień tygodnia (przekazywany jest on na zmiennej T, T = 1 — jeżeli jest to poniedziałek, 2 — jeśli wtorek,..., 7 — jeżeli niedziela).

```
700 R1 = INT (R/100)
710 R2 = MOD (R,100)
720 N = 2
730 IF M > 2 THEN 810
740 IF R2 = 0 THEN 770
750 IF MOD (R2, 4) = O THEN 800
760 GOTO 780
770 IF MOD (R1, 4) = 0 THEN 800
780 N = 4
790 GOTO 810
800 N = 3
810 T = MOD (INT (1.25 * R2)+INT (2.56 * M)+N+D,7)+1
820 RETURN
```

Znajdowanie daty k-tego w miesiącu j-tego dnia tygodnia

Algorytm ten pozwoli na znalezienie daty np. drugiej soboty listopada 1983, wówczas K = 2, J = 6, M = 11, R = = 1983. Jeżeli takiego dnia nie ma, to po wykonaniu procedury zmienna D ma wartość 0.

```
900 REM K =NUMER J—TEGO DNIA TYGODNIA
905 REM W MIESIĄCU M W ROKU R
910 REM D = SZUKANY NUMER DNIA
920 D = 1
930 GOSUB 700
940 D = 1+MOD (J+7—T,7)
950 D = (K—1) * 7+D
955 REM SPRAWDZENIE CZY OTRZYMANA DATA JEST POPRAWNA
960 D1 = D
970 M1 = M
980 R1 = R
990 GOSUB 100
1000 P1 = P
1010 D = 1
1020 M = M+1
1030 IF M < 13 THEN 1060
1040 M = 1
1050 R = R+1
1060 GOSUB 100
1070 IF P < P1+1 THEN 1120
1080 D = D1
1090 M = M1
1100 R = R1
1110 RETURN
1120 D = 0
1130 RETURN
```

Staraliśmy się wybrać najciekawsze algorytmy związane z Kalendarzem. Jeżeli ktoś z Czytelników zna krótsze lub bardziej interesujące procedury — prosimy o ich przesłanie.

ANDRZEJ SZAŁAS
ZBIGNIEW ŚWIRSKI

55 Międzynarodowe Targi Poznańskie

Rozwój rynku mikrokomputerowego

Tegoroczne Międzynarodowe Targi Poznańskie, które odbyły się w dniach 12—19 czerwca, dały wiele okazji do zastanowienia się nad możliwościami rozwoju nowoczesnej techniki komputerowej w dzisiejszych, niełatwych warunkach. Trudno podzielać oficjalny optymizm manifestowany w bieżących „targowych" doniesieniach, które ukazywały się w środkach masowego przekazu. W naszej dziedzinie nie było w zasadzie możliwości porównania krajowego sprzętu komputerowego z możliwościami choćby wytwórców europejskich, nie mówiąc już o osiągnięciach informatyki na świecie! Zresztą, podobnie było w latach ubiegłych. W tym roku przyjechało wprawdzie trochę firm zagranicznych, trudno jednak mówić o ofercie sprzętu informatycznego.

Przyjrzyjmy się zatem sprzętowi komputerowemu wystawianemu przez firmy z Polski i krajów RWPG. Zapoznajmy się z ofertą krytycznie, a na koniec podsumujmy fakty i wyciągnijmy wnioski.

ZSRR

Informatyka, podobnie jak dziedziny „pokrewne" (np. elektronika), miała niewielki udział w ogólnej ofercie ZSRR. Przedstawiono [1]:

● **Mikrokomputer typu SM 1300,** nowy model rodziny SM przeznaczony dla najmniejszych i najtańszych zestawów wykorzystujących tzw. „wspólną szynę". Jest on zgodny pod względem sprzęgu, a także programowo z SM 3, a przy tym co najmniej dwukrotnie wydajniejszy. Procesor realizuje podstawową listę rozkazów SM 3 i SM 4 ze wszystkimi trybami adresowania. Zbudowano go na układach scalonych rodziny KR 1802 (dominująca technologia — TTL LS). Długość mikrorozkazu — 48 bitów, czas wykonania rozkazu „R—R" — 2 μs. Pamięć operacyjna półprzewodnikowa, dynamiczna n-MOS, 64 K bajty × 18 bitów, o cyklu 700 ns. Mikrokomputer komunikuje się z operatorem przez monitor ekranowy, dzięki programowej emulacji pulpitu operatora z tradycyjnymi klawiszami i lampkami. Programy diagnostyczne wbudowano w pamięci ROM.

● **Dwumaszynowy zestaw** wykorzystujący wyżej opisane mikrokomputery **SM 1300** — przeznaczony do sterowania procesami, zbierania danych itp. Oprócz dwóch procesorów do podzielonej „wspólnej szyny" jest podłączony adapter do komunikacji tych procesorów oraz przełącznik szyny.

● **Komputer ES 1036.** Jest to nowy model maszyny Jednolitego Systemu; określa się go jako pierwszy stopień rozwoju klasy RIAD-3. Wykonuje 220 rozkazów zgodnych z wymaganiami JS. Wydajność — 400 tys. operacji na sekundę. Pamięć operacyjna — 2—4 M bajtów. Możliwa jest pamięć wirtualna, dynamiczne mikroprogramowanie, rozszerzone środki kontroli i diagnostyki. Jest pamięć buforowa „cache".

● **Komputer ES 1065.** Określano tę maszynę jako nowość, lecz nie udostępniono szczegółowych materiałów informacyjnych.

● **Pamięć na wymiennych dyskach magnetycznych ES 5066.** Dwunastodyskowy pakiet ma pojemność 100 M bajtów i jest w pełni zgodny z zaleceniami ISO. Gęstość zapisu 159 bitów/mm, średni czas dostępu — 55 ms. Jednostka sterująca ES 5566 wykonuje również testy i korekcję błędów.

BUŁGARIA

Ekspozycja Bułgarii była firmowana przez przedsiębiorstwo eksportowe ISOTIMPEX. Przed wejściem widniała reklama: „Computers Components Engineering", a spośród wystawionych urządzeń wypada odnotować:

● **Pamięć na małych dyskach elastycznych,** wykorzystującą nośnik o średnicy 5¼ cala, doskonale nadającą się do systemów mikrokomputerowych. Typ ES 5088 waży tylko 1,5 kg i wymaga zasilania napięciem stałym +12 V/0,9 A oraz +5 V/0,8 A. Zapis informacji odbywa się na jednej stronie dysku elastycznego na 35 ścieżkach, co daje pojemność pamięci 109,4 K bajty. Prędkość obrotowa dysku — 300 obr/min, pozycjonowanie głowicy trwa maks. 40 ms. Prędkość transmisji — 250 Kb/s.

● **System minikomputerowy ISOT 1016 C,** proponowany do zastosowań w dziedzinie sterowania procesami produkcyjnymi, automatyzacji prac doświadczalnych, systemów informacyjno-pomiarowych, zbierania danych oraz obliczeń naukowo-technicznych. Budowa systemu ISOT 1016 C jest modularna, a jego architektura opiera się o magistralę zwaną tu wspólną szyną (common bus, obščaja šina). Centralny procesor typu ISOT 2104 C (SM 2403) pracuje na 16-bitowych słowach i dysponuje 89 podstawowymi instrukcjami oraz 12 trybami adresowania. Może on zaadresować 256 K bajtów pamięci operacyjnej. Pamięć operacyjna ISOT 3501 C (SM 3501) jest pamięcią półprzewodnikową o maksymalnej pojemności 128 K słów (przyrost — co 4 K słów). Posiada czas dostępu 400 ns, czas cyklu — 700 ns i podtrzymanie zawartości — 1 godzinę po zaniku zasilania. System przerwań jest wektoryzowany, czteropoziomowy. Na każdym z poziomów można umieścić nieograniczoną liczbę przerwań. Jest stos zrealizowany sprzętowo. Oprócz licznych, typowych urządzeń zewnętrznych, wypada wymienić sterownik minidysków (do czterech jednostek o pojemności całkowitej 6 M bajtów każda) oraz sterownik małych pamięci taśmowych. Adapter międzysystemowy ISOT 4500 C pozwala na przyłączenie opisywanego minikomputera do maszyn ES przez typowy sprzęg w ich kanale.

NRD

Wystawa NRD-owskiej informatyki odbywała się — jak zwykle — pod znakiem firmowym ROBOTRON. Wystawiano urządzenia biurowe. Spróbujmy chociaż wyliczyć eksponaty:

● **Systemy do fakturowania i księgowania typu 1711 i 1355.**
● **System rozliczania sprzedaży A 5110.**
● **System planowania kosztów A 5120**
● **System gospodarki magazynowej A 5130.**
● **System przetwarzania danych dla centrum zarządzania A 5220.**

Pokazano również podzespoły do urządzeń informatyki:

● **Pamięć na minidyskach elastycznych typu 5600.10.**
● **Pamięć na kasetach magnetycznych typu 1152.**

CZECHOSŁOWACJA

Przedsiębiorstwo Handlowe KOVO przedstawiło:

● **Dalekopis elektroniczny CONSUL 231,** mający zastosowanie w sieciach teleksowych lub jako urządzenie zewnętrzne komputera. Nawiązuje do tradycji znanych dalekopisów elektromechanicznych produkowanych w przedsiębiorstwie Zbrojovka Brno. Jest produkowany w licznych wariantach. Układy elektroniczne tego dalekopisu są oparte o mikroprocesor typu 8080A. Program redakcyjny wraz z pamięcią użytkownika (typowo 2000 znaków) ułatwia odbiór oraz nadawanie wiadomości. Druk metodą mozaikową — matryca 10 × 7. Przesuw papieru i głowicy drukującej odbywa się za pomocą silników skokowych. Tekst nadawany i odbierany różnią się od siebie krojem pisma. Są duże i małe

[1]) Wszystkie dane zaczerpnięto z materiałów udostępnianych przez wystawców podczas Targów

litery. Prędkość pisania w ruchu miejscowym — 33 zn./s, prędkość telegrafowania — do 100 bodów. Dalekopis jest bardzo cichy, a jego obsługa łatwa i wygodna.

● **Mikrokomputer SM 50—40,** wykonany w oparciu o mikroprocesor typu 8080. Poszczególne moduły są połączone za pomocą sprzęgu 1—41. Oprócz typowych modułów, zawierających: mikroprocesor, pamięci operacyjne, programowalne sprzęgi we-wy, są jeszcze moduły sprzężenia z obiektem: wyjścia i wejścia dyskretne i analogowe, komutatory kanałów, liczniki, bloki przerwań itd. Jest zewnętrzna pamięć na dysku elastycznym. Przewidziano dwa wykonania: do wbudowania w inne urządzenia oraz do wbudowania w monitor ekranowy. Istnieje wersja ukierunkowana problemowo, zwana MVS-80, przeznaczona do wspomagania prac nad oprogramowaniem. Dostępne są dyskowe systemy operacyjne MUOS i MIKROS oraz egzekutor czasu rzeczywistego.

● **Mikrokomputer SM 50—50,** w pełni zgodny z SM 3/20 i SM 4/20. Lista rozkazów odpowiada SM 4/20, architektura wykorzystuje wspólną szynę, do której można dołączyć różne bloki, w tym urządzenia zewnętrzne. Przeznaczono ten mikrokomputer do sterowania procesami technologicznymi, do prac laboratoryjnych oraz do tworzenia terminali. Zbudowany jest na mikroprocesorze bipolarnym, segmentowym. Znane są dwie wersje: z adresowaniem pamięci do 64 K bajtów oraz rozszerzona — do 256 K bajtów. Odpowiednio do tego przewidziano różne pamięci półprzewodnikowe. Większa z nich ma słowa 16+6 bitowe, a mniejsza — 16+2 bitowe. System operacyjny, dyskowy FOBOS-2 jest w zasadzie systemem jednoużytkowym. Oprócz programów monitora, makroasemblera itp., dysponuje translatorami FORTRANU, kilku wersji BASICA, języka FOCAL i szeregiem programów pomocniczych. Rozszerzone wersje procesora mogą pracować z innymi systemami operacyjnymi, jak np. DOS-RVV2, DIAMS, DOS-RVR.

WĘGRY

Do najciekawszych eksponatów wystawionych przez węgierskie Przedsiębiorstwo Handlu Zagranicznego VIDEOTON należały:

● **Analizator mikrosystemów typu OL-622/1** — przenośny, walizkowy mikrokomputer z rodziny 8080. Przewidziany jest jako pomoc do nauczania oraz do prac prototypowych. Obsługa przyrządu — za pomocą prostej klawiatury. Magistrala systemowa wyprowadzona jest na złącze, podobnie jak i układy we-wy. Jest wyjście na magnetofon. Istnieje możliwość programowania pamięci PROM i EPROM, a także — emulacji.

● **Monitor ekranowy VDT 52127** do redagowania tekstów. Sterowanie mikroprocesorowe pozwoliło na dużą uniwersalność tego urządzenia, przy zachowaniu prostej obsługi. Ekran (12 cali) wyświetla 24 wiersze po 80 znaków w wierszu. Duże i małe litery, w tym cyrylica. Wśród kodów — ASCII, sprzęg V24 i pętla prądowa.

● **Przetwornik graficzno-cyfrowy** (koordynatometr, digimetr) rodziny RA. Informacja wyjściowa, zawierająca współrzędne punktów wykresu, może być przekazana za pomocą odpowiedniego sprzęgu do dziurkarki taśmy papierowej. Oprócz innych sprzęgów, istnieje też sprzęg szeregowo-asynchroniczny oraz BSI o poziomach TTL. Wymiary robocze stołu: 825 × × 525 mm lub 1125 × 750 mm. Graniczne dokładności: 1,25±0,2 mm lub 0,125±0,02 mm. Jest wyświetlacz współrzędnych.

● **Komputer biurowy VT 20/IV** — kontynuator znanego małego systemu cyfrowego VT 20A. Nowy system może jednocześnie obsługiwać czterech użytkowników, pracujących przy swoich inteligentnych terminalach. Cztery monitory ekranowe można uzupełnić drukarkami, są one podłączone do „jednostki centralnej", która ma do dyspozycji pamięć dyskową 2 × 2,5 lub 2 × 5 M bajtów. Typowa drukarka może drukować 300 wierszy na minutę. Można też zainstalować drukarkę mozaikową. Monitor ekranowy 12 cali wyświetla 24 wiersze po 80 znaków. Dość rozbudowany system operacyjny obejmuje m.in. relokowalny makroasembler, programy pomocnicze (np. edytor) oraz translatory:
— FORTRAN ANSI X 3.9—1966 z biblioteką programów
— interpreter BASICA z liczbami stało- i zmiennoprzecinkowymi oraz arytmetyką podwójnej precyzji, posiadający właściwości półgraficzne
— COBOL ANSI X3.23—1974
— PASCAL (jedna z wersji).

Całość pracuje w oparciu o mikroprocesory Z80A z zegarem 4 MHz. Należy podkreślić bardzo staranne opracowanie plastyczne systemu. Wydaje się, że była to jedna z nielicznych konstrukcji z krajów RWPG dorównująca pod wieloma względami przeciętnym opracowaniom z rozwiniętych krajów zachodnich, wyznaczających poziom w branży.

Ponadto Węgry, jako jeden z nielicznych krajów, oferowały oprogramowanie uzupełniające wcześniej opracowane systemy.

OFERTA KRAJOWA

Krajowa oferta informatyczna była dość różnorodna. Podzielmy ją zatem na kilka działów. Na początek — systemy.

● **System minikomputerowy MERA-SM.** Zakłady Wytwórcze Przyrządów Pomiarowych i Systemów Minikomputerowych w Warszawie pokazały system oznaczony jako MERA-CAMAC 125/SM 4A. Zestaw składa się z minikomputera MERA-SM oraz kasety CAMAC, ze sterownikiem typu 106A do sterowania tą kasetą. Właśnie blok typu 106A spełnia funkcję „interfejsu" zapewniając dwukierunkową wymianę informacji między minikomputerem a blokami systemu CAMAC, stanowiącymi tzw. sprzężenie z obiektem. Tak pomyślany system czasu rzeczywistego może być przeznaczony do automatyzacji procesów przemysłowych, badań jądrowych, laboratoriów naukowych, a nawet — dla medycyny. Wydaje się jednak, że opisywane rozwiązanie było tylko zręcznym pretekstem do oficjalnej prezentacji na Targach Poznańskich nowej [2]) rodziny maszyn cyfrowych — minikomputerów oznaczonych MERA-SM. Podstawowym elementem architektury tych minikomputerów jest tzw. wspólna szyna. Stanowi ona układ sprzęgu, do którego podłączone są wszystkie moduły systemu. Przesłanie informacji przez tę szynę o 56 sygnałach odbywa się asynchronicznie, na zasadzie tzw. uzgodnienia (ang. handshaking). Słowo maszynowe ma długość 16 bitów i może być traktowane jako dwa niezależne bajty. Bezpośrednia adresacja sięga 32 K słów, ale za pomocą jednostki zarządzania pamięcią można zaadresować 128 K słów. Urządzenia zewnętrzne traktowane są jako zbiory rejestrów o ustalonych adresach na „wspólnej szynie". Procesory mają osiem rejestrów uniwersalnych, w tym rejestr R7 pełni funkcję licznika rozkazów, a R6 — wskaźnika stosu. Lista rozkazów zależy od typu procesora i obejmuje rozkazy 0-, 1- i 2-adresowe; możliwych jest osiem trybów adresowania.

(Przeglądając materiały udostępniane przez producenta SM-ów przypomniałem sobie, że podobne rozwiązania opisywane są szeroko w licznych podręcznikach architektury komputerów. Zależność między rozwiązaniami „historycznymi" i „współczesnymi" pozostawiam domyślności i wnikliwości Czytelników.)

Najważniejsze oprogramowanie SM-ów to wielozadaniowy, wielodostępny system operacyjny czasu rzeczywistego — DOS RW. Ma on dwie odmiany: jako system „mapowany", z jednostką zarządzania pamięcią, oraz „niemapowany", na bazie 28 K słów pamięci. Udostępnia on użytkownikowi bogate oprogramowanie, m.in. FORTRAN, COBOL, PASCAL, BASIC-PLUS-2.

● **Rodzina mikrokomputerów serii 500.** Wrocławskie Zakłady Elektroniczne ELWRO przedstawiły nową generację swoich wyrobów: serię 500. Są to nowoczesne środki techniki biurowej, oparte na systemie mikroprocesorowym (8080). Przyjęto budowę modułową systemu. Część elektroniczna składa się z mikrokomputera jednopłytkowego, sterownika pamięci na dysku elastycznym, bloku monitora ekranowego i pamięci operacyjnej w modułach po 8 K bajtów. Urządzenie posiada klawiaturę, drukarkę ROBOTRON 1152, przystawkę do kart kontowych 1161 (też ROBOTRONU), dwie jednostki pamięci na dysku elastycz-

[1]) Producent udostępnia „Katalog Informatyki", Wyd. WEMA 1982, w którym nie występuje znana MERA-400; oprócz opisu MERY-SM, można tam znaleźć system numerycznego sterowania obrabiarką

nym PLx45D.5 oraz monitor telewizyjny wyświetlający 16 wierszy po 64 znaki. W oprogramowaniu można wyróżnić Podstawowy System Operacyjny z interpreterem języka ZIM, który jest podobny do języka asemblera oraz system operacyjny zgodny z dyskowym systemem operacyjnym CP/M.

● **Karty rejestracyjne CEKAR.** Centrum Komputeryzacji Rynku CEKAR przedstawiło rodzinę wielofunkcyjnych kas rejestracyjnych. Urządzenia te są nowoczesne dzięki zastosowaniu mikroprocesorów. Dane zapisywane do pamięci półprzewodnikowej są, oczywiście, chronione przy zaniku napięcia zasilającego. Omawiane kasy mają tak duże możliwości, że aż trudno je sobie wyobrazić w warunkach naszego handlu. Np. kasa rejestracyjna CEKAR 212 jest przeznaczona jako pomoc do zarządzania średnimi i dużymi sklepami samoobsługowymi lub prowadzącymi sprzedaż tradycyjną. Umożliwia ona m.in.: programowanie ceny jednostkowej dla 100 rodzajów towarów identyfikowanych dwucyfrowym numerem, programowanie ceny jednostkowej w czterech lub siedmiu grupach towarowych oraz rejestrowanie utargu i liczby operacji w grupie, rejestrowanie wypłat w czterech grupach, rejestrowanie obrotu w trzech formach płatności, rejestrowanie utargu i obrotu w każdej z form płatności — jednostki pamięci z małym dysjednego lub dwóch kasjerów, wyznaczanie i rejestrowanie narzutów procentowych, rejestrowanie utargu i obrotu dziennego oraz miesięcznego dla całej kasy, drukowanie raportów informacyjnych (podglądowych) i końcowych itd. Ponadto kasa ta może być wyposażona w układy transmisji danych. Oczywiście, są i prostsze rozwiązania, np. kasa rejestracyjna CEKAR 12 przeznaczona do małych sklejów o tradycyjnej formie sprzedaży. Kasa CEKAR 222 jest przeznaczona dla barów, kawiarni itp. Z kolei terminal kasowy CEKAR 100 CR/S posiada odpowiednio zwiększone możliwości rejestracji, a ponadto można do niego dołączyć wagę, urządzenie do wydawania reszty, a opcjonalnie: pamięć zbiorów, rejestrator danych, monitor lub drukarkę, wreszcie — czytnik kodu paskowego (ang. bar code).

● **Sterownik mikroprocesorowy MIKROSTER.** Zakład Urządzeń do Montażu Podzespołów Elektronicznych ze Szczytna wystąpił jako producent opracowanego w Przemysłowym Instytucie Elektroniki modułowego sterownika mikroprocesorowego MIKROSTER [3]). Bloki tego systemu, przeznaczone do automatyzacji procesów produkcyjnych, oznaczone są jako MSA-80.xx. Dziś można mówić o 25 standardowych modułach—pakietach. Pakiety te mają wymiar 140 × 150 mm oraz złącze 84-stykowe, dzięki któremu mają dostęp do równoległej magistrali systemowej. Należy jednak przypuszczać, że magistrala ta nie odpowiada żadnemu ze znanych standardów światowych. Oprócz pakietów procesora, oferowano pakiety pamięci, pakiety sprzężenia z urządzeniami peryferyjnymi oraz do bezpośredniej współpracy z obiektem. W podobny sposób zbudowany jest mikroprocesorowy system automatyzacji pomiarów MSP-80. Oba wykorzystują układy scalone z rodziny 8080. Całość uzupełnia system projektowo-uruchomieniowy MIKROSTER, posiadający trzy wersje: MSA-80. 1000 — najprostsza; MSA-80. 2000 — z oprogramowaniem rezydentnym; MSA-80. 3000 — z dyskowym systemem operacyjnym.

Przedsiębiorstwa Polonijno-Zagraniczne uczestniczyły we wspólnej ekspozycji. Zdumiewa znaczny wzrost ilościowy tych przedsiębiorstw; rok temu było ich kilkanaście, teraz ponad 80. Wprawdzie większość z nich oferuje towary typowo konsumpcyjne (konfekcja, kosmetyki itp.), to są i takie, które sprzedają elementy niezbędne do produkcji innych dóbr. Właśnie firmy wytwarzające sprzęt elektroniczny i urządzenia informatyki stanowią wśród tych ostatnich dużą, wyodrębnioną grupę. (Główne produkty tych przedsiębiorstw przedstawia w drugiej relacji z Targów Janusz Zalewski — przyp. red.)

Warto ponadto omówić przedstawione w Poznaniu różne urządzenia, zwłaszcza zewnętrzne oraz inne elementy niezbędne do kompletowania systemów cyfrowych.

● **Jednostki pamięci na dyskach elastycznych** — wystawiła Krakowska Fabryka Aparatów Pomiarowych. Na tle znanej jednostki typu PLx45.D.5, która miała dwa dyski ośmiocalowe zapisywane jednostronnie z pojedynczą gęstością zapisu, co dawało 400 KB pojemności, proponuje się:

— jednostki pamięci z małym dyskiem ośmiocalowym, typu FD 802 D, z zapisem dwustronnym i podwójną gęstością zapisu, co daje 1,6 MB; liczba ścieżek — 76, szybkość transmisji — 500 kHz, średni czas dostępu — 220 ms; Interface Shugart SA 851, nośnik 200 mm, format DIS 7065

— jednostki pamięci z małym dyskiem elastycznym (5¼), typu E 501 D, E 501 D, E 502 D; zapis jest dwustronny z pojedynczą i podwójną gęstością, w jednostce 501 — jednostronny (nie wynika to jasno z ulotki); maksymalna pojemność — odpowiednio 250 i 500 KB; liczba ścieżek — 40, szybkość transmisji — 125/250 kHz, średni czas dostępu — 200 ms; sprzęg ANSI, dysk o średnicy 130 mm (tj. 5¼) o formacie DIS 7487; wymiary: tylko 85 × × 150 × 220 mm!, masa: tylko 1,5 kg!

● **Drukarki mozaikowe** — wystawiły Zakłady Mechaniczno-Precyzyjne w Błoniu. Pokazano trzy typy: D-200, D-180 i D-100.

Drukarka D-200 jest szeregową drukarką z rastrem 7 × 9 punktów i szybkością drukowania 180 zn/s. Jest sterowana mikroprocesorem typu 8035. Druk może odbywać się w obu kierunkach z gęstością 10, 12 lub 16,5 znaków/cal — poziomo, a pionowo — 6, 8 lub 10 wierszy/cal. Papier perforowany lub nie, o szerokości od 4 do 17 cali. Bufor znaków — 1 wiersz. Posiada ona wiele rodzajów druku: normalny, pochyły, szeroki, szeroko-pochyły, wąsko-wysoki, szeroko-wysoki. Te rodzaje druku mogą wystąpić w kilku postaciach: normalnej, o podwójnej intensywności i wyrazistej. Brak interlinii pozwala na druk wykresów metodą kropek, druk kodów paskowych (bar code) oraz druk semigraficzny.

Drukarka D-180 drukuje z prędkością 180 zn/s i 40—50 linii/min. Znak 7 × 9 punktów, 132 lub 158 kolumn. Bufor 256 znaków. Sprzęg — CENTRONICS, kod ASCII, poziomy logiczne — TTL. Możliwe są inne sprzęgi.

Drukarka D-100 jest przedstawicielem drukarek trzeciej generacji. Jest mała i lekka: wymiary 410 × 320 × 120 mm i masa 12 kg! Sterowanie mikroprocesorowe. Głowica drukująca 9-igłowa. Napędy za pomocą silników skokowych. Można stosować papier w rulonie nieperforowany i oddzielne arkusze o szerokości do 8,5 cala, a ponadto — papier obrzeżnie perforowany o szerokości od 4 do 9,5 cala. Można drukować maks. 80 znaków w wierszu przy gęstości 10 zn./cal i szybkości druku 100 zn./s. Sprzęgi: IRPR (DZM-180), S2 (V-24), CENTRONICS. Kod — ASCII.

Pamięć wewnętrzna półprzewodnikowa PWP-256/22 — wystawiły ją Zakłady Wytwórcze PP i Systemów Minikomputerowych. Jest przeznaczona dla minikomputerów ze „wspólną szyną". Dzięki wyposażeniu pamięci we własne źródła zasilania, stanowi ona jednostkę autonomiczną. W razie zaniku zasilania sieciowego wewnętrzna bateria akumulatorów umożliwia zachowanie informacji w pamięci przez ok. 1 godzinę. Pamięć PWP-256/22 jest nowoczesną pamięcią półprzewodnikową o zwiększonej niezawodności: MTBF nie mniej niż 10 tys. godzin. Pamięć posiada układy autokorekcji ECC, eliminujące pojedynczy błąd w słowie i sygnalizujące błędy w większej liczbie bitów. Pojemność pamięci wynosi 128 K słów 16-bitowych oraz 6 bitów kontroli ECC. Odczyt słowa (i bajtu) trwa maks. 700 ns, zapis słowa — 700 ns, ale zapis bajtu — 1200 ns, Odświeżanie 700 ns — co 15,6 µs. Zastosowano układy scalone o pojemności 16 K bitów.

● **Analizator stanów logicznych E-220** — przedstawiło Przedsiębiorstwo Doświadczalno-Produkcyjne Elektronicznej Aparatury Pomiarowej EUREKA. Jest to pierwsza od lat poważna oferta przyrządu tego typu, tak niezbędnego do uruchamiania i napraw sprzętu cyfrowego. Sposób zapisu danych — synchroniczny lub asynchroniczny z możliwością wykrywania pojedyn-

[3]) Szczegółowe materiały informacyjne rozpowszechnia Przemysłowy Instytut Elektroniki, Branżowy Ośrodek Informacji Naukowej, Technicznej i Ekonomicznej; ul. Marynarska 10, 02-674 Warszawa; tel. 47-08-12; teleks 815530 pie pl. Informacji techniczno-handlowych udziela w ww. Instytucie — Dział Marketingu i Wdrożeń; ul. Długa 44/50, 00-241 Warszawa; tel. 31-51-55, teleks 813260 pie pl. (por. INFORMATYKA. nr 2/1983)

czych wąskich impulsów. Posiada osiem wejść danych, zegar oraz jedno wejście triggera. Zegar od 20 MHz do 50 Hz, wyzwalanie 8-bitowym słowem. Pamięć 256 słów 8-bitowych, opóźnienie wyzwalania: 0—9999 okresów zegara. Wyświetlanie na oscyloskopie, monitorze ekranowym lub telewizorze.

Zagadnienia informatyki były prezentowane także przez placówki naukowe i szkolnictwo wyższe. Przykładowo, Politechnika Warszawska przedstawiła: Modułowy System Mikroprocesorowy MSM (por. INFORMATYKA, nr 2/1983), komputerowy system analizy statycznej STRAINS-80 oraz zastosowanie systemu MERA 9150 w dydaktyce.

Na koniec — niespodzianka: Zdobyłem reklamówkę pierwszej w kraju prywatnej firmy informatycznej! Firma BIO-RYTM z Wrocławia oferuje rozwiązywanie problemów z dziedziny statystyki, obliczeń inżynierskich, obliczeń ekonomicznych, badań operacyjnych, optymalizacji oraz — poradnictwo obliczeniowe i pomoc przy podejmowaniu decyzji. Ponadto Pierwsza w Kraju Prywatna Firma Informatyczna — jak głosi reklama w pięciu językach — oferuje jako narzędzie ostatnią zdobycz — „dźwignię intelektualną".

* * *

Wszystkie minikomputery 16-bitowe oferowane przez kraje RWPG są oparte o „wspólną szynę" i przedstawiają zbliżony poziom techniczny. Pozostaje teraz czekać na rozwój mikroprocesora 16-bitowego. Dziś jeszcze go nie widać, ale postęp techniki prowadzi ku niemu.

W dziedzinie mikrokomputerów 8-bitowych jest wiele propozycji. Oprócz oficjalnie uznanego mikroprocesora typu 8080, w wielu rozwiązaniach pojawia się uparcie mikroprocesor Z80A. Warto przypomnieć, że w NRD jest produkowany jego odpowiednik — U 880 D, wraz z kilkoma innymi układami scalonymi z tej rodziny. Ciekawe, jak na dzisiejszej sytuacji układów scalonych do systemu mikroprocesorowego zaważy szansa naprawdę masowego ich wykorzystania. Szansę tę dają kasy rejestracyjne dla handlu oraz — w pewnym stopniu — mikrokomputery biurowe i inżynierskie. No cóż, zobaczymy.

Obrót oprogramowaniem — zwłaszcza podstawowym — jest dość ściśle związany z dostawą sprzętu. Czy jeden wyjątek może być dziś zapowiedzią zmian na przyszłość? Choć sprawa jest ważna, chyba jeszcze poczekamy na małe, niezależne firmy zajmujące się oprogramowaniem. Obecnie małe firmy (PPZ) skupiają uwagę raczej na sprzęcie.

W dziedzinie urządzeń peryferyjnych odczuwa się wkroczenie minidysków elastycznych. Oferowane są przez Polskę, NRD, Bułgarię. Sygnalizowane jest też pojawienie się nowoczesnych, małych drukarek. Widać już różne monitory ekranowe z telewizorem. Gdyby to wszystko można było bez większych problemów kupić, wówczas warto byłoby robić nowoczesne mikrosystemy. Właśnie przed nimi jest przyszłość. Powinny coraz bardziej ważyć w rozwoju krajowej informatyki.

JACEK ŻEBROWSKI

55 Międzynarodowe Targi Poznańskie

Komputery firm polonijno-zagranicznych

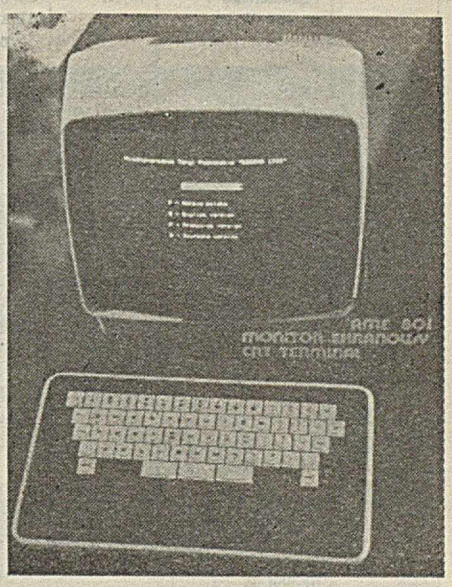
Z zainteresowaniem jechałem do Poznania, gdzie firmy polonijno-zagraniczne miały przeprowadzić mikrokomputerową ofensywę. Zapowiedzią tej ofensywy były już choćby ubiegłoroczne Targi.[1] A pierwsze spotkane stoiska handlowe potwierdziły te oczekiwania.

Firma MARCO ELECTRONICS z Krakowa przedstawiła multiplekser MUX 04+ do mikrokomputera APPLE II, oparty na mikroprocesorze typu 8085A. Dzięki uzyskanym w ten sposób czterem dodatkowym kanałom transmisji asynchronicznej, wg normy V24, możliwe jest utworzenie — w oparciu o ten mikrokomputer — 5-terminalowego systemu wielodostępnego. Multiplekser pełni rolę wysuniętego procesora wykonującego wszystkie czynności związane z transmisją (maksymalna szybkość transmisji — 4800 bodów). Od strony mikrokomputera jest widziany jako dodatkowa pamięć typu RAM o pojemności 256 bajtów. Bufor o tej pojemności odpowiada każdemu z czterech dodatkowych terminali (fot. 1). Wykorzystuje się system operacyjny DOS z własnymi modyfikacjami wieloużytkowymi.

Rozszerzeniem jest także konwerter M-20/60, przystosowujący urządzenia transmisji danych, wg normy V24, do wymagań pętli prądowej 20 lub 60 mA, dzięki czemu zasięg transmisji zwiększa się do 10 km. Wiodącym wyrobem tej firmy staje się multiplekser 32-kanałowy, umożliwiający terminalom lub innym urządzeniom asynchronicznym, zgodnym z normą V24, wspólny dostęp do kanału transmisji danych. Czas transmisji przydziela się tylko terminalom aktywnym. Obowiązuje protokół ASP (ang. Addressing Serial Protocol), który jest zmodyfikowanym protokołem DDCMP stosowanym przez firmę DEC. Modułowa konstrukcja urządzenia umożliwia łatwe rozszerzanie zestawu do 32 kanałów.

Ogólnie — wyroby tej firmy stwarzają duże możliwości w automatyzacji prac biurowych.

O ile firma MARCO ELECTRONICS w zasadzie nie reklamowała komputera, to na innym stoisku mikrokomputer reklamował się sam, bez firmy. Okazało się, że grupa młodych konstruktorów opracowała mikrokomputer gotowy do seryjnej produkcji (pełna dokumentacja) i poszukuje sponsora. Ich dzieckiem jest komputer osobisty o nazwie AGAT (fot. 2), oparty na procesorze Z80A, z możliwością adresowania obszaru pamięci



Fot. 1. Terminal współpracujący z komputerem za pomocą multipleksera firmy MARCO ELECTRONICS

[1]) J. Żebrowski: 54 Międzynarodowe Targi Poznańskie Inwazja mikroprocesorów. INFORMATYKA, nr 7/1982, str. 28

do 1 M bajtu (przez układ zarządzania pamięcią). Do komputera, przez opracowane sterowniki, można dołączyć prawie wszystkie urządzenia zewnętrzne spotykane w tej klasie sprzętu, tzn.: monitor ekranowy (CRT 8275), stację dysków elastycznych, pamięć kasetową PK-1, perforator i czytnik taśmy papierowej oraz drukarkę, modem i urządzenia analogowe. Istnieje również możliwość dołączenia sterownika graficznego (1024 × ×1024 punkty) i kilkunastu terminali. Obecnie komputer może pracować pod nadzorem systemów operacyjnych CP/M i ISIS. Przewidywany jest własny system operacyjny z językiem BASIC. Mikrokomputer może znaleźć zastosowanie głównie w laboratoriach i pracowniach przemysłowych i naukowo-badawczych. Dlatego przewiduje się, że tzw. wejścia—wyjścia obiektowe, jak również oprogramowanie użytkowe będzie powstawać raczej na zamówienia indywidualne.

Ciekawym rozwiązaniem jest modem oparty na konstrukcji modułowej, umożliwiający tworzenie złożonych połączeń terminali z komputerem. Dotychczas skonstruowano następujące moduły: zasilacz, nadajnik, odbiornik i sterownik, a przewidywane są dalsze, np. do współpracy z telefonem, teleksem, linią telefoniczną itd. Transmisja odbywa się synchronicznie z szybkością 1200 lub 2400 bitów/s. Twórcy komputera są pewni, że jego cena będzie konkurencyjna nie tylko w stosunku do wyrobów firm państwowych, ale także — polonijnych.

Komputerem osobistym już dobrze znanym czytelnikom INFORMATYKI [2] jest ZX 81, montowany i sprzedawany w Polsce przez firmę AMEPROD z Poznania, przedstawiony także w bieżącym numerze w artykule J. Karczmarczuka. Warto natomiast dodać, że konstruktorzy tej firmy opracowują mikrokomputer o większych możliwościach, zwany K-100P (fot. 3), który będzie uzupełniony o klawiaturę, grafikę, analogowe wejście—wyjście itp. Wydaje się, że dzięki względnie tanim wyrobom i masowej sprzedaży, firma ma już ustaloną pozycję na rynku polskim, zaczyna nawet wydawać biuletyn użytkowników ZX 81.

Jednak ZX 81 jest zabawką w porównaniu z systemem mikrokomputerowym CS-80 firmy COMPUTEX (fot. 4). Jest to system modularny, oparty o mikroprocesor ZX 80A, przeznaczony do zastosowań laboratoryjno-przemysłowych, konstrukcyjno-biurowych, a także typowo obliczeniowych. Dotychczas skonstruowano szereg pakietów, m.in. pakiety jednostki centralnej różniące się pojemnością zamontowa-

[2]) J. Dawidowski: Komputer osobisty ZX 81. INFORMATYKA nr 5/1983, str. 21

Zestawienie mikrokomputerów firm polonijno-zagranicznych

Nazwa mikrokomputera	Dostawca	Procesor	Pojemn. pamięci (obszar adres.)	Sterowniki urządzeń zewnętrznych	Magistrala	Wymiar płytek (mm)	Złącza magistrali	Oprogramowanie	Ceny (zł)	Poziom informacji (1—5)
1)	MARCO ELECTRONIC Balicka 176-190 30-149 Kraków	I-8080A	do 64KB	wszystkie typowe + multiplekser asynchron.						(2,5)
AGAT	2)	Z80A-4 MHz	do 0,5 MB (1 MB)	dyski elast. monitor kol. V24 PK-1 we-wy anal.	specjalna	100 × ×160		Zgodne z CP/M i ISIS przewidywany własny syst. oper. i BASIC		2
ZX-81	AMEPROD Kmieca 20a 61-632 Poznań	Z80A-3,25 MHz	do 64KB	klawiatura membranowa, monitor drukarka magnetofon				BASIC	228 000 (64 KB + druk.)	3
CS-80	COMPUTEX Skrońskiego 7 02-466 W-wa	Z80 Z80A	EPROM do 8 KB: dRAM do 64KB	monitor ekr. dyski elast. we-wy cyfr. we-wy anal. monitor graf.	zunifikowana	100 × ×240	64 stykowe	sys. op. MAGOS Control BASIC DEBUGGER	250 000 (MAGOS)	4
MSM	IMPOL Malawskiego 7 02-641 W-wa	I-8080A I-8085A Z80A	EPROM do 16KB: sRam-do 16KB	PLx45D PK-1 monitor ekr. we-wy anal. we-wy cyfr.	BUSMAT	100 × ×160	pośrednie 64 st. CANNON	monitor sys. op. MOPS BASIC KAMIL sys. op. DOPS 3)	30 000 59 000 90 000 93 000 159 000 150 000 + K10000	4
MSK-1	IMPOL 1 Malawskiego 7 02-641 W-wa	INTEL 8080	EPROM do 16KB: dRAM do 64KB	DT-105 CT2200 DZM 180 dyski elast. we-wy cyfr. we-wy anal.	drukowana	220 × ×223,4	811/821 ELTRA	monitor + debugger edytor tek. makroasemb. disasembler 3)	47 000 42 000 75 000 42 000 178 000 + K10000	3,5
IMZ-80	IMPOL II Filona 16 02-658 W-wa	Z80	do 64KB	monitor ekr. magnetofon DZM 180 PL-45D PK-1				BASIC asembler edytor debugger		2
IMP-85	IMPOL II Filona 16 02-658 W-wa	INTEL 8085A	do 64KB	monitor ekr. PLx45D DZM 180 V24				sys. op. IMPS BASIC 80 FORTRAN80 makroasemb.		2

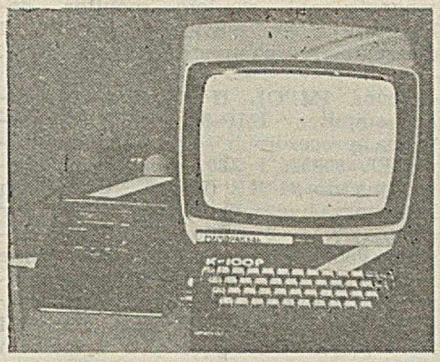
1) Dokładne informacje u dostawcy. 2) Informacje w firmie DEKOR, Forteczna 11, 01-540 Warszawa. 3) Procesor + RAM

Fot. 2. Komputer osobisty AGAT

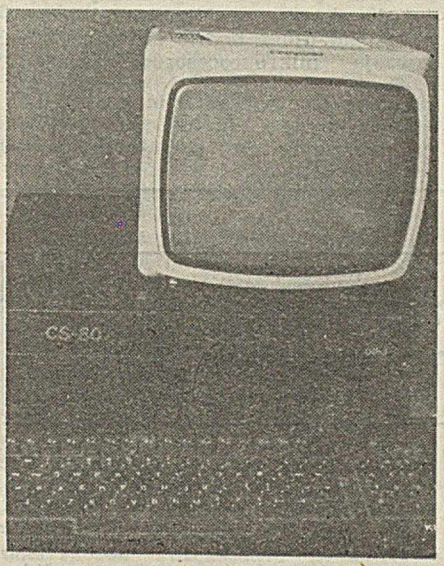
nej pamięci, sterowniki monitora ekranowego (24 × 40 lub 24 × 80 znaków) i graficznego (512 × 256 punktów), pakiety wejść—wyjść cyfrowych i analogowych oraz sterownik dysków elastycznych.

Oprogramowanie występuje w trzech wersjach różniących się nośnikiem informacji: taśma papierowa, taśma magnetyczna (pamięć kasetowa PK-1) i dysk elastyczny (stacja SP45DE). W opracowaniu jest także system z dyskietką 5-calową. W skład każdej wersji oprogramowania wchodzi system operacyjny MAGOS, interpreter języka BASIC i program uruchomieniowy DEBUGGER (zgodny ze standardem INTELA). MAGOS jest wzorowany na systemie operacyjnym CP/M i rezyduje w pamięci EPROM zajmując ok. 6 KB obszaru adresowego. Do podstawowych poleceń systemu należą LIST, KILL, PRINT, STORE, NAME, LOAD i EXEC, których nazwy tłumaczą funkcje. Na życzenie klienta MAGOS jest dostarczany w formie całkowicie zgodnej z systemem CP/M. Podstawowy język programowania, tzw. CONTROL-BASIC, jest szybkim interpreterem, opartym na składni BASICA, ukierunkowanym na pracę w czasie rzeczywistym. Umożliwia pracę wielozadaniową dzięki takim instrukcjom, jak: TASK, ACTIVATE, TURNON, WAIT, DISMISS, CLOCK itd., przy czym wymiana informacji między zadaniami zachodzi tylko przez wspólną pamięć.

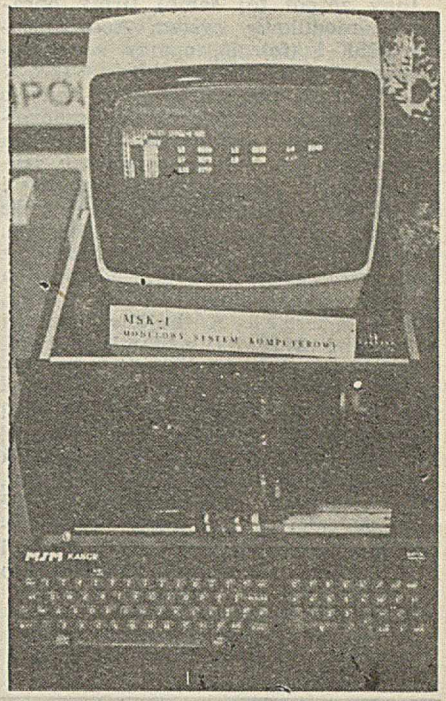
Równie uniwersalny jest modułowy system mikroprocesorowy MSM firmy IMPOL 1, umożliwiający tworzenie zestawów pomiarowo-sterujących, obliczeniowych, dydaktycznych, komputerów biurowych itp. Jednostki centralne są oparte na mikroprocesorach I-8080A, I-8085A lub Z80A. Moduły w zestawie łączą się za pośrednictwem magistrali BUSMAT (dokładniejszy opis w katalogu), która umożliwia pracę wieloprocesorową, adresowanie 1 MB pamięci i przesyłanie 10 sygnałów przerwań. Wymiary pakietów odpowiadają pojedynczej eurokarcie, wyposażonej w 64-stykowe złącze pośrednie. Obszerny zbiór modułów



Fot. 3. Wersja robocza mikrokomputera K-100P firmy AMEPROD



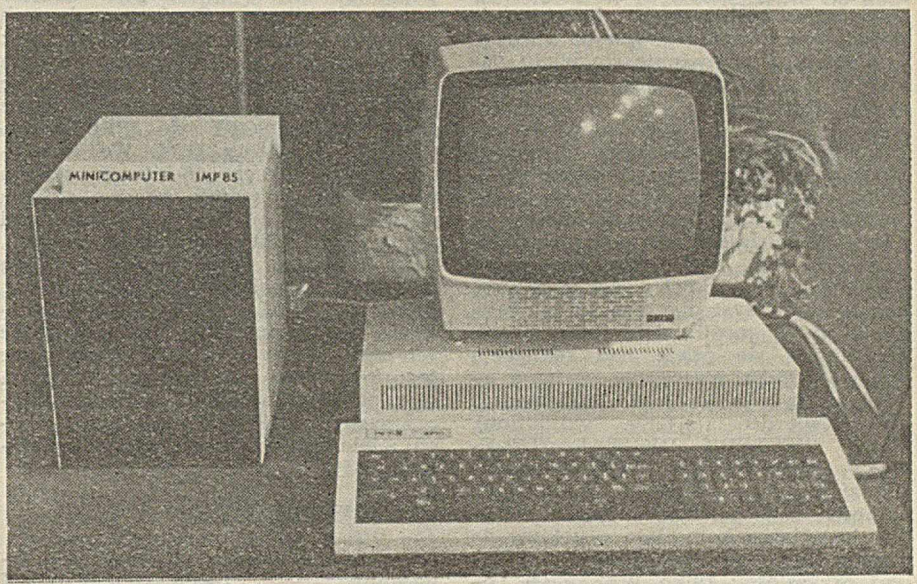
Fot. 4. System mikrokomputerowy CS-80 firmy COMPUTEX



Fot. 5. Modułowy system komputerowy MSK-1 firmy IMPOL I

obejmuje sterowniki pamięci kasetowej i dyskietkowej, monitora ekranowego, układy wejść—wyjść cyfrowych szeregowych i równoległych oraz — analogowych. Interesująca jest możliwość rozszerzenia magistrali i dołączenia drugiej kasety.

Oprogramowanie obejmuje prosty monitor, minimalny system operacyjny MOPS lub dyskietkowy system operacyjny DOPS, zgodny z ISIS II, interpretery języków BASIC i KAMIL do sterowania sekwencyjnego (brak danych nt. składni.).



Fot. 6. Mikrokomputer IMP85 firmy IMPOL II

Inny zespół tej samej firmy opracował modułowy system komputerowy MSK-1 (fot. 5), oparty konstrukcyjnie także o eurokartę (220 × × 233,4 mm), przeznaczony do szeroko rozumianej automatyzacji w przemyśle, medycynie, handlu itd. Magistrala stanowiąca drukowane połączenie szyny danych, adresowej, sterującej i zasilania, wykorzystuje górne złącze każdego modułu (ELTRA 811/812). Dolne złącza są używane zgodnie z funkcją każdego modułu. Jednostka centralna, oparta na rodzinie INTEL 8080, zawiera 6 K bajtów pamięci EPROM i 2 K bajty pamięci RAM. Opracowano lub opracowuje się sterowniki typowych urządzeń zewnętrznych produkowanych w Polsce, jak: perforator DT 105, czytnik CT 220, drukarka DZM-180, pamięć kasetowa PK-2 i pamięć dyskietkowa z KFAP, a ponadto — układy wejść—wyjść analogowych i cyfrowych. Oprogramowanie, dostarczane w pamięci EPROM, na taśmie perforowanej lub magnetycznej, zawiera monitor, debugger, edytor tekstowy, asembler i disasembler oraz programator EPROM.

Firma IMPOL II przedstawiła mikrokomputery IMP-85 i IMZ-80 oparte na procesorach — odpowiednio — INTEL 8085A i Z80, z pamięcią operacyjną do 64 KB (fot. 6). Zestawy są lub mogą być wyposażone w monitor ekranowy, 1—2 jednostki dysków elastycznych, drukarkę DZM-180 i inne urządzenia zewnętrzne. Oprogramowanie obejmuje system operacyjny IMPS zgodny z CP/M oraz BASIC i inne języki działające pod systemem CP/M, a także — podstawowe oprogramowanie narzędziowe.

* * *

Jak wynika z tego omówienia, oferta firm polonijno-zagranicznych jest już dosyć pokaźna. Nie ma tylko pewności, jak długo będzie aktualna. Choć inwazja mikroprocesorów postępuje nadal, to jest ukierunkowana jednak raczej na aparaturę niż na klienta. Należy przy tym przyznać, że większość zestawów jest atrakcyjna tak ze względu na możliwości techniczne, jak i ceny. Nie widziałem, niestety, żadnej dokumentacji, ale jest to chyba powszechna słabość.

Firmy polonijno-zagraniczne są z pewnością pożądanym elementem na polskim rynku komputerowym, wypełniając pewną lukę produkcyjną, a rozmach niektórych (np. IMPOL) wróży perspektywę ich trwałej obecności. Zresztą, niektóre mikrokomputery, np. CS-80, MSM, MSK, uzupełnione o już produkowany, specjalny sprzęt, mogą z powodzeniem konkurować — jako zestawy uruchomieniowe — z systemami wspomagania przedstawionymi w pierwszym tegorocznym numerze INFORMATYKI.

JANUSZ ZALEWSKI
(Fot. KACPER)

Oprogramowanie mikrokomputerów

W ubiegłym roku dominującym czynnikiem w rozwoju oprogramowania było dążenie do zwiększenia produktywności programistów, zamierzone jako przeciwwaga ich ciągłego niedoboru. Zwiększenie produktywności stanowi podstawowy cel nowych opracowań, jak np.: przystosowanie 8-bitowych systemów operacyjnych do procesorów 16-bitowych, definiowanie standardów programowych do różnych zastosowań i pojawienie się wielu pakietów do automatyzacji oprogramowania użytkowego.

Z punktu widzenia rynku mikroprocesorowego najbardziej znaczące są — zapowiedziane przez wytwórców oprogramowania (ang. software houses) — adaptacje ważniejszych systemów operacyjnych dla mikrokomputerów 16-bitowych. Przewidując, że te mikrokomputery i systemy operacyjne będą miały duże możliwości obliczeniowe, kontynuuje się prace nad realizacją nowego języka wysokiego poziomu — ADA, przybliżając jego implemencje do zatwierdzenia (ang. validation). Firmy WESTERN DIGITAL, TELESOFT, SUPERSOFT i INTEL zapowiedziały wprowadzenie na rynek kompilatorów podzbioru języka ADA. Ale kompletne kompilatory gotowe do zatwierdzenia nie są sprawą najbliższej przyszłości; tylko WESTERN DIGITAL podał, że jest gotów przekazać swoją wersję do testowania procesu, który zacznie się prawdopodobnie pod koniec roku, po przekazaniu programu zatwierdzającego (ang. validation program) przez Departament Obrony.

Wśród wytwórców systemów operacyjnych istnieje tendencja do adaptowania systemów 8-bitowych na maszyny 16-bitowe. Oprócz czołowych systemów uniwersalnych, takich jak: CP/M, Unix i system programowania UCSD (UCSD p-system), pojawił się system MS-DOS dostarczany przez firmę MICROSOFT do PERSONAL COMPUTER firmy IBM. We wszystkich przypadkach 16-bitowy system operacyjny zachowuje pewien stopień wymienności z dobrze określonymi 8-bitowymi systemami operacyjnymi. Przy tworzeniu systemu CP/M-86 dla 16-bitowego procesora INTEL 8086, firma DIGITAL RESEARCH stwierdziła, że system może wykorzystywać ten sam format danych, jaki użyto dla jego 8-bitowego poprzednika. Ta sama wymienność formatów danych dotyczy systemu MP/M-86, 16-bitowej wersji wielodostępnego systemu operacyjnego MP/M.

ZACHOWANIE ZGODNOŚCI

Opierając się na istniejących rozwiązaniach — zarówno system UNIX z Bell Laboratories, jak i UCSD p-system są zaimplementowane na maszynach 16-bitowych, odpowiednio — za pomocą kompilatora języka C lub interpretera kodu programu. System MS-DOS ma wewnętrzną strukturę podobną do UNIXA i zapewnia przejście do systemów 16-bitowych przez emulację odwołań systemowych CP/M. W ten sposób 16-bitowe systemy operacyjne zapewniają zwiększoną produktywność programistów, dzięki zgodności (ang. compatibility) z zaakceptowanymi systemami 8-bitowymi. Zarówno dostawcy, jak i użytkownicy uznali ten pomysł za istotny, ponieważ ułatwia użytkowanie sprzętu wysokiej jakości.

Wiele systemów jest zbudowanych w oparciu o istniejące oprogramowanie 8-bitowe, dla procesorów obu grup: 8- i 16-bitowych. Przykładowo — system RAINBOW firmy DIGITAL EQUIPMENT używa mikroprocesorów Z80 i 8088, a system COMPUPRO/ /SYSTEMS — podwójnego procesora łączącego układy typu 8085 i 8088. Systemy takie są przystosowane do rozszerzania oprogramowania na maszyny 16-bitowe. System operacyjny CP/M-86 dla komputera RAINBOW posługuje się pojedynczym formatem danych przy ściąganiu programów z dysku i wykonuje je na odpowiednim procesorze. System operacyjny MP/M-8/16 dla komputera firmy COMPUPRO spełnia podobne funkcje w środowisku wielodostępnym i wielozadaniowym.

W systemie programowania UCSD firmy SOFTECH MICROSYSTEMS wprowadzone zostało pojęcie kodu programu na poziomie pośrednim (p-code). Umożliwia ono użytkownikom tworzenie oprogramowania przenośnego. Kod wynikowy (p-code object code) programu użytkowego, wytworzo-

ny i skompilowany na jednym procesorze, np. na LSI-11, może być wykonany bezpośrednio na innym, np. Z8000, przez rezydujący interpreter p-kodu. Chociaż szybkość obliczeń, w porównaniu z maksymalną możliwością do uzyskania, jest mniejsza, to jednak wytwórcy i użytkownicy uznali, że do zastosowań ogólnych jest ona wystarczająca. W ten sposób system programowania UCSD można uważać za jedyny system operacyjny wspólny zarówno dla PERSONAL COMPUTER firmy IBM, jak i dla DISPLAYWRITER SYSTEM firmy SOFTECH.

Oprogramowanie pracujące w czasie rzeczywistym stawia systemom operacyjnym szczególnie wysokie wymagania. Systemy oparte na procesorach Z8000 i M68000 są wspomagane przez oddzielne wersje systemu operacyjnego MSP firmy HEMENWAY, natomiast procesory iAPX 86 i 88 INTELA mogą używać własnych systemów RMX-86 lub RMX-88. Przenosząc właściwości systemów operacyjnych czasu rzeczywistego pracujących na dużych komputerach, systemy te zapewniają szeregowanie zadań, wprowadzają mechanizmy przydziału zasobów i komunikacji — wymagane w pracy wielozadaniowej. Podstawą do tworzenia oprogramowania pracującego w czasie rzeczywistym są programy zarządzające do obsługi transakcji przypadków (ang. event-driven transactions). Dla projektantów programów użytkowych istotne zaś jest to, że każdy system jest otoczony dodatkową warstwą oprogramowania uruchomieniowego (ang. development software) — edytorami, kompilatorami, programami ładującymi przeznaczonymi do tworzenia kodu wynikowego oraz programami uruchomieniowymi pracującymi na bieżąco (ang. real-time debugger).

INTEL uczynił krok naprzód w projektowaniu programów użytkowych, bazujących na systemie operacyjnym RMX-86 — definiując zbiór komórek standardowych w systemie operacyjnym. Wykorzystywanie tych komórek podczas projektowania i pracy zapewnia, że pakiety oprogramowania użytkowego mogą być przenoszone do następnej generacji mikrosystemu spełniającego wymagane normy. Podobnie, opracowane przez firmę INTEL rozszerzenie systemu RMX — tzw. Multibus Message Exchange — jest używane w systemach wieloprocesorowych.

W większości systemów operacyjnych, które nie są projektowane specjalnie do realizacji zadań w czasie rzeczywistym, względy wydajności nie odgrywają znaczącej roli. Niemniej wzrastająca liczba dostępnych sprzętowych jednostek zarządzania pamięcią wymaga systemów operacyjnych umożliwiających obsługę (ang. handle) pamięci wirtualnej. Projektanci tych systemów będą zmuszeni zająć się konkretną sprzętową implementacją zarządzania pamięcią: w jednym przypadku będzie to segmentacja, jak np. w układzie INTEL 80286 mającym w układzie jednostkę zarządzania pamięcią, w innym — żądanie stronicowania, jak np. w układzie jednostki zarządzania pamięcią firmy NATIONAL, przystosowanym do współpracy z procesorem typu 16032. Analogicznie, rozpowszechnienie specjalizowanych sterowników do dysków i terminali spowoduje wzrost wymagań stawianych podprogramom sterowania (ang. driver) wejścia—wyjścia, używanym przez system operacyjny — zmuszając prawdopodobnie do dostosowania ich do nowego sprzętu.

Bardziej obiecującym kierunkiem rozwoju, z punktu widzenia projektanta systemów operacyjnych, jest dążenie do stworzenia systemu uniwersalnego, który używałby zbioru dobrze zdefinowanych komórek standardowych. W sytuacji idealnej komórki te służyłyby do połączenia jądra systemu operacyjnego z kodem zależnym od sprzętu. Chociaż upłynie zapewne sporo czasu, zanim systemy zbudowane warstwowo będą używane powszechnie, to jednak tendencja do ich wprowadzania jest już widoczna (od kiedy zaczęto stosować standardowe sprzęgi). Przykładowo — system MS-DOS firmy MICROSOFT używa warstwowego protokołu zobrazowania (ang. presentation level protocol driver), tak samo jak nowy układ sterownika graficznego firmy INTEL.

Żaden pojedynczy system operacyjny nie może stać się standardem 16-bitowym, ponieważ każdy ma inne możliwości: to, co jest niezbędne dla wielodostępnych systemów operacyjnych, może być niepotrzebne w systemie operacyjnym komputera osobistego przeznaczonym dla jednego użytkownika. Z tego względu firma INTEL także popiera utworzenie sprzęgu standardowego od strony użytkowej systemu operacyjnego. Jego zadaniem jest skuteczne oddzielenie programów użytkowych od obszarów podległych systemowi operacyjnemu. Sprzyja się w ten sposób zwiększeniu produktywności programistów posługujących się tą techniką.

Zaproponowane przez INTELA sprzęgi, tzw. Universal Development Interface (UDI) i Universal Run-time Interface (URI) definiują zbiór komórek systemowych zbliżony do Microprocessor Operation System Interface, propozycji standardu opracowanej przez Institute of Electrical and Electronics Engineers (IEEE). UDI i URI określają obecnie komórki dla pięciu obszarów zastosowań: pamięci, plików, zarządzania procesami, sterowania sytuacjami wyjątkowymi i wspomagania zastosowań (ang. application assistance). Wykorzystując te komórki programista ma możliwość przenoszenia swoich podprogramów do dowolnego systemu operacyjnego, bez względu na to, jaki jest główny procesor.

Tzw. krzemowe systemy operacyjne (ang. silicon operating systems), jak np. Versalite Real-Time Executive (VRTX) firmy HUNTER AND READY, wspomagają implementację programów użytkowych pracujących w czasie rzeczywistym w środowisku wielozadaniowym. Komórki systemu VRTX mogą być używane przez projektantów oprogramowania — tak jakby były zwykłym rozszerzeniem listy rozkazów procesora. System VRTX udostępnia komórki systemowe dla sześciu różnych obszarów zastosowań: sterowania zadaniami, komunikacji i synchronizacji, wprowadzania—wyprowadzania znakowego, zarządzania zegarem i pamięcią oraz sterowania obsługą przerwań.

OPROGRAMOWANIE UNIWERSALNE

Firma SOFTECH MICROSYSTEMS usiłuje położyć kres nadmiarowi formatów dyskowych, prowadzącemu do kłopotów związanych z inwentaryzacją. Rozwiązanie proponowane przez SOFTECH umożliwia różnym jednostkom sterującym czytanie tego samego dysku za pośrednictwem tzw. Universal Medium (nośnika uniwersalnego). Dane są zapisywane na dysku w formacie pojedynczym, który jest stosowany w większości dysków. Pakiet użytkowy (ang. utility package) lub adapter programuje sterownik dysku tak aby akceptował on format standardowy, następnie tłumaczy dane zapisane w postaci standardowej na format wymagany przez konkretny system. Opracowana przez SOFTECH koncepcja Universal Medium, akceptowana już przez dostawców pamięci masowych, jest podobna do koncepcji pośredniego kodu programu (ang. intermediate p-code) tej firmy i prowadzi do tego samego rodzaju przenośności między dyskami za pomocą programowanych sterowników i adapterów programowych.

Koncepcja kodu pośredniego jest stosowana także w tzw. Universal Language System firmy PHILON GROUP. W tym przypadku kompilatory są pisane tak, by generowały kod pośredni (q-code), a nie instrukcje maszynowe określonego procesora. Użycie takiego kodu na różnych maszynach wymaga takiego samego postępowania, jak z kodem pośrednim w systemie programowania UCSD: asembler określonego procesora wytwarza kod maszynowy dla procesora docelowego, wykorzystując kod pośredni jako daną wejściową. W przeciwieństwie do koncepcji UCSD system firmy PHILON (q-code) jest przeznaczony dla producentów sprzętu OEM i kosztuje ok. 100 tys. dolarów.

Koncepcja kodu pośredniego zwiększa produktywność umożliwiając przenoszenie pakietów oprogramowania użytkowego. Innym podejściem do tego zagadnienia jest, będące w początkowym etapie rozwoju, automatyczne generowanie programów. Automatyczne generatory programów służą do wstępnego przetwarzania przy współpracy z bazami danych — umożliwiają nie-programistom posługiwanie się wyszukanym raportowaniem lub służą jako sprzężenie z bibliotekami podprogramów — i używają zbiorów podprogramów użytkowych w celu utworzenia zwykłego programu.

Lista dostępnych automatycznych generatorów programów zawiera ok. 40 pozycji i stale się powiększa, w związku z dużym zapotrzebowaniem na oprogramowanie użytkowe. Następną generacją automatycznych generatorów programów będą prawdopodobnie systemy sztucznej inteligencji (ang.

expert systems, knowledge-based systems), które pomogą użytkownikowi w sformułowaniu problemu, a następnie odwzorują uzyskany zestaw połączeń logicznych w odpowiednie kroki obliczeniowe. Należy się spodziewać, że w bieżącym roku pojawią się na rynku pewne odmiany systemów samoorganizujących się, które będą występować pod nazwą „automatycznych generatorów programów".

SYSTEMY WSPOMAGANEGO PROJEKTOWANIA

Z punktu widzenia oprogramowania, projektowanie wspomagane komputerowo (ang. computer aided design, CAD) należy do dziedziny szybkich przekształceń matematycznych. W ubiegłym roku stwierdzono, że nadszedł już czas na konsolidację narzędzi automatycznego projektowania w systemy mogące opracowywać finalne projekty układów scalonych na podstawie opisu funkcjonalnego. Systemy te mogą przy tym dostarczać testów układu (ang. test patterns). Urządzenia pomocnicze i symulatory układowe są względnie powszechne, lecz następny krok w rozwoju systemów CAD zostanie zrobiony w kierunku większego uzależnienia wyników od systemu. Projektanci sprzętu będą mogli naśladować metody robocze projektantów oprogramowania, tworząc kompletne opisy funkcjonalne i opisy działania systemu, a pomijając przy tym szczegóły wykonania czy nawet — konstrukcji układu.

Jednym z niebezpieczeństw grożących układom scalonym, ignorowanym zwykle przez projektantów, jest rozpraszanie ciepła, szczególnie w dużych, energochłonnych konstrukcjach. Do rozwiązywania tego zagadnienia jest przeznaczony zespół urządzeń CAD firmy HITACHI nazywany Hisets. Określa on statyczny i dynamiczny rozkład temperatury w poprzek wielowarstwowej i różnorodnej struktury, na podstawie opisu sieciowego źródeł ciepła.

Oprac. JACEK WALAS

na podstawie
ELECTRONICS z 20 października 1982

Mikrokomputer na co dzień

Obecnie we Francji można kupić średniej klasy mikrokomputer za miesięczną lub dwumiesięczną pensję inżyniera. Nastąpił przy tym gwałtowny rozwój zastosowań. Z jednej strony — zwiększa się znacznie pole dotychczasowych zastosowań w przemyśle, handlu, administracji itp. (nawet najmniejsze kilkuosobowe przedsiębiorstwo może już sobie pozwolić na zakup mikrokomputera), z drugiej — mikrokomputer coraz częściej kupowany jest przez osoby prywatne jako jeszcze jeden element wyposażenia mieszkania, służący do zabawy lub prac domowych.

Następuje wyraźne rozszerzenie się zakresu zastosowań mikrokomputerów — poza obszar informatyki profesjonalnej. Zarazem jednak można zaobserwować ruch niejako przeciwny, związany z zawodowym użytkowaniem mikrokomputerów — początkowo pomyślanych jako urządzenia dla odbiorcy masowego, nie związanego z informatyką.

Masowa produkcja i sprzedaż mikrokomputerów spowodowała rewolucję w systemie ich sprzedaży. Oczywiste jest, że istniejąca sieć handlowa producentów, nastawiona na sprzedaż pojedynczych komputerów, nie może sprostać nowym wymaganiom rynku. Powierzono zatem sprzedaż mikrokomputerów dużym wyspecjalizowanym domom towarowym lub całym sieciom takich domów, obejmujących swym zasięgiem terytorium danego kraju. Sklepy z mikrokomputerami nie różnią się w istocie od tych z telewizorami czy aparatami radiowymi. Na marginesie — agencje handlowe IBM we Francji nie przyjmują zamówień na mikrokomputery w liczbie niższej niż 20 sztuk!

*

Mikrokomputer jest komputerem zbudowanym w oparciu o mikroprocesor lub mikroprocesory i jest to jedyne kryterium nie podlegające dyskusji. Jeśli przyjmiemy, że obecnie produkowane mikrokomputery mają np. pamięć operacyjną wielkości od 0,12 KB do 1 MB, że mogą nie mieć żadnej pamięci zewnętrznej lub że można przyłączyć do nich pamięć dyskową rzędu ponad 100 MB, zauważymy łatwo, że mamy do czynienia z charakterystykami komputerów średniej mocy.

Najczęściej spotykana obecnie na rynku konfiguracja mikrokomputera przedstawia się następująco:
— jednostka centralna z pamięcią operacyjną 32 lub 64 K, z możliwością rozbudowy do 128 lub 256 K
— monitor ekranowy 24×80 znaków z klawiaturą
— dwie jednostki pamięci na dyskach elastycznych
— drukarka znakowo-mozaikowa o szybkości 120 znaków/s.

Z reguły spotyka się w mikrokomputerach procesory 8-bitowe. Coraz liczniejsze są już jednak procesory 16-bitowe, a pojawiły się też 32-bitowe.

Najczęściej spotykane obecnie rozwiązanie konstrukcyjne to: monitor ekranowy z wbudowaną jednostką centralną i jednostkami pamięci na dyskach elastycznych, niezależna klawiatura oraz drukarka (firma APPLE). Spotyka się też inne rozwiązania: jednostka centralna z pamięcią na dyskach elastycznych jako jeden element oraz osobno ekran, klawiatura i drukarka (firma IBM).

Stosowane są również rozwiązania w postaci jednego urządzenia, w którym (w jednej obudowie) znajdują się: ekran, jednostka centralna, pamięci na dyskach elastycznych, klawiatura i drukarka.

Typowym rozwiązaniem stosowanym w „domowych" mikrokomputerach jest możliwość przyłączenia do zwykłego telewizora i wykorzystywania go jako monitora ekranowego oraz magnetofonu kasetowego jako pamięci zewnętrznej.

*

W odróżnieniu od stosunkowo niewielu producentów komputerów czy minikomputerów, liczba producentów mikrokomputerów jest znaczna. W samych Stanach Zjednoczonych jest ich ponad 150, a w Japonii ponad 70. Jeśli dodamy do tego producentów europejskich nikogo już nie zdziwi, że niemal każdy tydzień przynosi kilka nowych konstrukcji.

Dla lepszego zobrazowania podstawowych charakterystyk dzisiejszych mikrokomputerów podaję zestawienie (tab. 1), w którym za podstawowe kryterium klasyfikacyjne przyjęto cenę. W tabeli podane są ceny rynkowe, które płaci prywatny nabywca. W przypadku zakupu inwestycyjnego dokonywanego przez przedsiębiorstwo — ma ono prawo do zwrotu podatku, który wynosi 18,6%. Tak więc praktyczna cena jest wówczas znacznie niższa.

W tabeli 2 podano mikrokomputery określane jako osobiste — ze względu na stosunkowo niską cenę (nie przekraczającą 20 000 FF, czyli poniżej 3000 dol.). W tabeli tej można odróżnić dwie podstawowe grupy.

Pierwsza — sprzęt w cenie od 670 FF (ZX 81) do 4000 FF (NEW BRAIN), który ze względu na cenę można określić jako nieprofesjonalny; sprzedawany jest masowo w sklepach i domach towarowych. Funkcję monitora spełnia tu ekran telewizora.

Druga — w cenie 16,5—20 tys. FF to już praktycznie komputery profesjonalne, zarówno ze względu na cenę, jak i konfigurację. Z reguły mają

Tabela 1. Porównanie mikrokomputerów 16-bitowych

	IBM PC	DIGITAL RAINBROW 100	THOMSON MICROMEGA 32	OLIVETTI M20	RADIO SHACK TRS MODEL 16	SIRIUS 1
Mikroprocesor	8088	8088 i Z 80 (8 bitów)	MC 68000	Z 8001	MC 68 000 i Z 80 (8 bitów)	8088
Pojemność pamięci operacyjnej (maksymalna)	512 KB	256 KB	256 KB—1 MB	224 KB	512 KB	896 KB
Monitor ekranowy	Biało-czarny lub kolorowy 25×40/80 znaków	Biało-czarny lub kolorowy 24×80/132 znaków	Jednobarwny 25×80 znaków	Jednobarwny lub kolorowy 24×80 znaków	Jednobarwny 24×80 znaków	Jednobarwny 25×80/50 znaków 50×132 znaków
Klawiatura	QWERTY odłączalna	QWERTY odłączalna	AZERTY odłączalna	QWERTY lub AZERTY	QWERTY	AZERTY lub QWERTY
Dyski elastyczne	2 dyski 5 1/4" 2×163 KB lub 2×600 KB	2 dyski 5 1/4" 2×400 KB	od 1 do 4 dysków 5 1/4" o pojemności 1 MB każdy	2 dyski 5 1/4" 2×320 KB	4 dyski 8" 4×1,25 MB	2 dyski 5 1/4" 2×600 KB lub 2×1200 KB
Dyski sztywne	ew. 2 ×19 MB	ew. 1 Winchester 5 1/4" 5 MB	od 1 do 4 dysków 5 1/4" — od 5, 10 do 20 MB każdy	ew. 9 MB	ew. 4×8,4 MB	—
Sprzęgi	ew. RS 232C	RS-232C lub RS-422	V 24, RS-232C lub IEEE 488	RS-232C oraz równoległy	2×RS-232C	ew. IEEE 488
Jednostanowiskowy	Tak	Tak	Tak	Tak	Tak	Tak
Wielostanowiskowy	Nie	Nie	Tak	Nie	Tak	Nie
System operacyjny	MS-DOS lub CP/M86 i UCSDp	CP/M86	UNIX — LIKE	P-COS	UNIX — LIKE	CP/M86 lub MS-DOS
Cena wersji podstawowej	25 000 (z pamięcią 64 KB)	32 000 (z pamięcią 64 KB)	65 200	36 500 (z pamięcią 128 KB)	44 950	29 900 (z pamięcią 128 KB)

Tabela 2. Wybrane komputery osobiste

Model	Producent	Cena (FF)	Pamięć wewnętrzna		Ekran				Pamięć zewnętrzna		Drukarka	Uwagi
			ROM (KB)	RAM (KB)	telewizor czarno-biały	ekran czarno-biały	telewizor kolorowy	ekran kolorowy	kasetowa	dyskowa		
ZX 81	SINCLAIR	670	8	1	×				×			w częściach 490 FF
VIDEOJEU C 52	PHILIPS	1 320	1	0,3			×					mikrokomputer do gier
V C S	ATARI	1 490	4	0,12			×					do gier
INTELLIVISION	MATTEL	1 980	1,2	6	×		×		×			do gier
ZX 81	SINCLAIR	2 016	8	16	×				×		+	
ZX SPECTRUM	SINCLAIR	2 050	16	16	×		=		×	+		
TI 99/44	TEXAS INSTRUMENTS	2 950	26	16			=				×	
T07	THOMSON	3 750	16	8			=		×	×		
NEW BRAIN	GRUNDY	3 978	29	32	×	•			+	× =		• ekran 1×16 znaków
TX 25	CANON	16 500	24	32		•			×	+		• ekran 1×20 znaków
GOUPIL 2	SMT	16 814	6	58		+	=		×	=		
TRS 80 model 1	TANDY	17 575	12	32		+			+	+ +		
OSBORNE 01	OSBORNE	19 000	4	64		+				+ +		
APPLE II PLUS	APPLE	19 381	12	64		+				+ + =		
AVT II	AVT	19 950	12	64		+			×	+ + =		

• Legenda pod tabelą 3

one własny monitor ekranowy i dwa dyski elastyczne.

W tabeli 3 podano natomiast mikrokomputery w cenie od 25—75 tys. FF. Najczęściej mają one dwie dyskietki o pojemności od 100 KB do 1,2 MB każda oraz możliwość przyłączenia dysku typu WINCHESTER o pojemności najczęściej 5 lub 10 MB (w tabeli, w rubryce 9, cyfry z prawej strony oznaczają pojemność jednego dysku elastycznego).

Wiele dzisiejszych mikrokomputerów ma procesory 8- i 16-bitowe, jak np. RAINBOW 100 firmy DIGITAL EQUIPMENT czy TRS 80 Model 16 firmy TANDY. Przy tym pamięć operacyjna może być rozbudowana do pojemności ponad 512 KB (APPLE II PLUS, SIRIUS 1, PEP 2000).

Przy porównywaniu cen należy zwrócić uwagę na fakt, że niektórzy producenci w zestawie podstawowym nie uwzględniają drukarki (znak „+" w rubryce 13), a więc do ceny tych mikrokomputerów należy doliczyć koszt drukarki (ok. 8000 FF).

*

Przewidywanie przyszłości zawsze było i jest niebezpieczne, jednak w dziedzinie mikrokomputerów wiele elementów rozwojowych można przewidywać z dużą dozą pewności. O tej przyszłości mówią najlepiej barwne i nieco przesadzone tytuły w fachowych

Tabela 3. Wybrane mikrokomputery „profesjonalne"

Model	Producent	Cena (FF)	Mikroprocesor		Pamięć operacyjna		Ekran		Klawiatury AZERTY	Pamięć zewnętrzna	System operacyjny			Drukarka w zestawie
			8 bitów	16 bitów	minimum	maximum	czarno-biały	kolorowy			CP/M lub MP/M	MS/DOS	Inne	
GOUPIL 2	SMT	25 395	+		58	58	+	×	+	++= 98			FLEX	
M 23	SORD	25 873	+		128	128	+		+	++= 328			MDOS	
HP 85 A	HEWLETT-PACKARD	26 055	+		32	32	+			×= 270			HP	+
CBM 4000	COMMODORE	27 048	+		32	32	+			++ 170			COMMODORE	+
APPLE II PLUS	APPLE	27 781	+		12	512	+	×		++ 140			DOS 3.3	+
HZ 80 B	SHARP	28 946	+		66	66	+			×××× 256			SHARP	
ITT 3030	ITT	29 189	+		64	256	+		+	++= 280	+			
XEROX 820	RANK-XEROX	30 070	+		64	64	+		+	++= 256	+			
MC 1	R2E	31 013	+		64	64	+		+	++= 600	+			
RAINBOW 100	DIGITAL-EQUIPMENT	31 634	+ 8088		64	256	+	×	+	++= 400	+			
ALFATRONIC	TRIUMPH-ADLER	32 855	+		48	64	+		+	++= 160			MOS	+
MBC 4050	SANYO	33 200		8086	128	512	+		+	++= 640	+			
BX 3	CANON	34 700	+ 8088		64	128	+			++= 320			MCX	+
S 1	VICTOR	35 162		8080	128	512	+		+	++= 600	+	+		
SANCO 800	LOGIC-SYSTEM	35 256	+		70	198	+		+	++= 800	+			
M 20	OLIVETTI	36 238		Z8001	128	284	+		+	++= 286			P.COS	
LX 528	LOGABAX	36 648	+		64	64	+		+	++= 320	+			+
CEM 800	COMMODORE	40 984	+		32	96	+		+	++= 500			COMMODORE	+
PERSONAL COMPUTER	IBM	43 512		8088	64	512	+			++ 640	+	+		
APPLE III	APPLE	44 546	+		128	256	+		+	++= 140	×		SOS	+
ENTERPRISE 100	DATA GENERAL	48 552	micro nova		64	64	+		+	++= 356			MP/OS	
PEP 2000	PEP	50 000	+		64	512	+			++= 320	+		PEP DPS	+
QUESTRAM/M	CII-HB	53 796	+		62	62	+		+	++= 256			Q/MOS	
WANGWRITER	WANG	54 332	+		96	128	+		+	++= 300	+			+
IPC M30	ICL	56 933	+		64	512	+		+	++D 256	+			
TRS 80 model 16	TANDY	58 564	+ 68000		128	512	+			+×=1200			TRS DOS	
BASF 7120	BASF	59 600	+		88	88	+		+	++= 160			BOS	
HP 125	HEWLETT-PACKARD	64 591	+		64	64	+			++= 256	+			+
RAINROW 100	DIGITAL-EQUIPMENT	69 148	+ 8088		64	256	+		+	++D 400	+			
PROFESSIONAL 350	DIGITAL-EQUIPMENT	75 382		F-11	128	128	+		+	++D 400			POS	

Legenda:

+ — występuje w zestawie
× — opcja
D — dysk sztywny 5 MB
= — dysk sztywny w ramach opcji
(tab. 2 — sprzęg inny niż SECAM)

czasopismach francuskich: „LISA prawie tak prosta jak władanie piórem i kartką papieru", „OLIVETTI liczy na sprzedaż 60 000 mikrokomputerów w 1983 roku", „1000 mikrokomputerów GOUPIL 3 sprzedano w dwa miesiące", „Pierwszy mikro 16-bitowy z pamięcią 64 K w cenie poniżej 10 000 FF", „Jeszcze jeden mikro od wielkiego producenta", „Gorączka mikrokomputerowa", „Wojna mikrokomputerów zbliża się".

Na marginesie „wielkiej wojny" pomiędzy czołowymi producentami sprzętu komputerowego można zauważyć, że w przeciwieństwie do monopolu w dziedzinie produkcji komputerów, zarezerwowanego dla krajów wysokouprzemysłowionych, w dziedzinie mikrokomputerów wiele krajów Trzeciego Świata podejmuje już ich produkcję w oparciu o import podzespołów i elementów. Jest to niewątpliwie nowy element. Czytelnika INFORMATYKI może — na przykład — zaskoczyć informacja, że w Brazylii istnieje już obecnie około trzydziestu producentów mikrokomputerów. Podobne działania obserwuje się w innych krajach Trzeciego Świata.

W chwili obecnej kluczowym elementem decydującym o rozwoju rynku mikrokomputerowego jest możliwość szybkiej realizacji prostych zastosowań, a więc masowa sprzedaż gotowego oprogramowania. W Stanach Zjednoczonych programy do mikrokomputerów sprzedawane są dzisiaj w... księgarniach, gdyż sieć firm zajmujących się sprzedażą oprogramowania i usługami informatycznymi jest niewystarczająca dla obsługi wielomilionowej już rzeszy użytkowników.

Wszystkie te czynniki powodują bardzo dynamiczny wzrost sprzedaży mikrokomputerów. W roku 1983 w samych tylko Stanach Zjednoczonych przewiduje się sprzedaż ok. 3 mln mikrokomputerów! Rynek mikrokomputerowy, który osiągnął już obroty roczne rzędu 6 mld dol., w ciągu najbliższych trzech lat uzyska prawdopodobnie ponad trzykrotny wzrost. Jest się więc nad czym zastanowić.

MICHAŁ ZIĘBIŃSKI

Słowniczek terminów związanych z systemami graficznymi

Niniejszy słowniczek stanowi uzupełnienie serii artykułów o systemach graficznych prezentowanych w INFORMATYCE (nr 3—9/1983). Terminologia polska nie jest w tej dziedzinie jednolita i ugruntowana, a sytuację pogarsza jeszcze fakt, że również w angielskojęzycznej literaturze fachowej używane są różne terminy na oznaczenie tego samego pojęcia. Stąd trudności w podawaniu określeń równoważnych. Zestawienie podstawowych terminów w formie słowniczka może pomóc w studiowaniu publikacji angielskojęzycznych omawiających systemy graficzne.

W jednym z następnych numerów rozpoczniemy dokładniejsze omawianie terminologii grafiki komputerowej z uwzględnieniem aktualnych norm międzynarodowych (ISO i SIGGRAPH).

cyfrowe przetwarzanie obrazów	— digital image processing (digital) picture processing
graficzne urządzenie wejściowe	— graphical input device interaction device
czytnik rysunków	— digitizer digitizing tablet (data) tablet
klawiatura (funkcyjna)	— (function) keyboard
manipulator drążkowy	— joystick
manipulator kulowy	— trackball crystal ball tracker ball
grafika komputerowa	— computer graphics
grafoskop	— vector system, vector display refresh line-drawing display random-scan display stroke display calligraphic display
kadr	— frame
element obrazu	— pixel picture element pel point
linia	— line
półobraz	— field
kineskop	— CRT Cathode Ray Tube
lampa pamięciowa	— DVST (Direct View Storage Tube)
terminal graficzny z lampą pamięciową	— storage-tube display
pamięć obrazowa	— image memory framebuffer, framestore refresh buffer display refresh memory
płaszczyzna pamięci obrazowej	— image memory plane bit plane
dodatkowa płaszczyzna pamięci obrazowej	— overlay (graphics) overlay plane
percepcja wzrokowa	— visual perception
barwa	— hue
kolor	— am. color ang. colour
luminancja	— brightness
nasycenie	— saturation
poziom szarości	— gray level
pisak xy	— (digital) plotter
bębnowy	— drum
elektrostatyczny	— electrostatic
stołowy	— flatbed
rastrowy system graficzny	— raster (graphics) system raster display raster-scan display
sterownik wizyjny	— video generator display controller
tablica pośrednia	— LUT (Look-Up Table) video look-up table color table color map intensity transformation unit (image) function memory
układ dyskretyzacji sygnału wizyjnego	— frame grabber video image digitizer video data converter TV image digitizer
całkowity sygnał wizyjny	— composite video signal
urządzenie trwałych kopii	— (output) hardcopy device
wybieranie kolejnoliniowe	— non-interlaced (scanning)
wybieranie międzyliniowe	— interlaced (scanning)
wygaszanie pionowego powrotu plamki	— field blanking
pionowy powrót plamki	— vertical retrace
wygaszanie poziomego powrotu plamki	— line blanking
poziomy powrót plamki	— horizontal retrace
znacznik	— cursor

JACEK OWCZARCZYK

W USA ogłoszono, że firma IBM zakupiła za 250 mln nowych akcji (12%) firmy INTEL — jednego z przodujących producentów półprzewodników. Transakcja sankcjonuje kierunek, jaki pojawił się w przemyśle półprzewodnikowym kilka lat temu: niezależni wytwórcy układów scalonych, których produkty są na szczytowym poziomie technologicznym, poszukują dopływu gotówki wraz z pewnego rodzaju partnerstwem ze strony wielkich międzynarodowych koncernów. Koszty badań i rozwoju stały się tak wysokie, a zyski tak duże, że tego typu porozumienia są nieuniknione. Współpraca IBM-INTEL rozpoczęła się wcześniej. Przy użyciu mikroprocesora INTEL 8088 zbudowano IBM PERSONAL COMPUTER, później podpisano kontrakt na dostawę dynamicznych układów RAM 64 K. INTEL rozwija w swej produkcji technologię CMOS (np. procesor 80386, nowe pamięci 64 K), a IBM chce oprzeć na niej swoje wyroby i jest również zainteresowany w stosowaniu pamięci domenowych, które wytwarza INTEL. Bezpośrednie udziały w INTELU pozwolą utrzymać się IBM w czołówce technologicznej, zwiększając jednocześnie konkurencyjność swych wyrobów, z czym nie było najlepiej w ciągu ostatnich 2—3 lat. Z kolei dla firmy INTEL kontrakt ten oznacza potężny zastrzyk gotówki. Jednocześnie firma w pełni zchowuje swoją niezależność — w umowie ograniczono udziały IBM do 30% w ciągu ośmiu lat (co w mikroelektronice oznacza trzy generacje układów). Według obserwatorów amerykańskich, kontrakt ten wskazuje na coraz większe zmiany w sposobie postępowania IBM, który zaczyna częściej sięgać po podzespoły innych producentów. Wiąże się to jednocześnie z przebudową działalności rynkowej giganta, co przypisuje się obecnemu prezesowi Johnowi R. Oplowi.

*

Centrum Naukowe IBM France zaprezentowało oficjalnie stan zaawansowania swoich prac w dziedzinie rozpoznawania kształtów i systemów typu EKSPERT. Stosunkowo długo przedsięwzięcia tego typu, określane zwykle hasłem sztucznej inteligencji, uważano za domenę ośrodków uniwersyteckich i za godne zainteresowania wyłącznie ze strony teoretyków. Jednak obecnie stały się one przedmiotem intensywnych prac nad zastosowaniami praktycznymi i to w tradycyjnie niechętnej nowinkom firmie IBM. Przypuszcza się, że pierwsze systemy będą służyły diagnostyce chorób nowotworowych oraz modelowaniu obiektów trójwymiarowych.

*

Ponad 500 tys. osób ze stu krajów odwiedziło w okresie 13—19 marca 1983 wiosenne Targi Lipskie. Widoczną tendencją tegorocznej edycji targów było prezentowanie zastosowań mikroelektroniki we wszelkich sektorach gospodarki. Niezależnie od tego, jeden z 22 salonów specjalistycznych poświęcono informatyce. Uczestniczyli w nim wystawcy z 14 krajów i Berlina Zachodniego. Stosunkowo bogatą ofertę przedstawiły kraje socjalistyczne. VEB KOMBINAT ROBOTRON (NRD) pokazał 35 nowych wyrobów, w których zastosowano mikroelektronikę, a wśród nich system ROBOTRON A 6471/6373 do przetwarzania obrazów oraz terminal bankowy G 8924. Rumunia zaprezentowała wyniki swej współpracy z firmą CONTROL DATA — jednostkę dyskową (RCD 9747) 60 M bajtów i czytnik kart (RCD 9228). Podzespoły do tych urządzeń są wytwarzane w USA i Rumunii, montowane we Frankfurcie (RFN) i stamtąd eksportowane na cały świat. Związek Radziecki oferował nowy monitor ekranowy (zbudowany w oparciu o mikroprocesor TS 7966) wyświetlający jednocześnie teksty angielskie i rosyjskie oraz drukarkę Romaszka posiadającą jednocześnie zbiór znaków łacińskich i cyrylicę.

*

Po raz pierwszy w historii języków programowania normalizacja wyprzedziła praktyczną realizację. 17 lutego br. American National Standard Institute (ANSI) zatwierdził normę języka ADA, wyprzedzając o cztery miesiące zakończenie prac nad pierwszym kompilatorem. Proces ten poprzedziły dwie rundy konsultacji, podczas których zebrano ok. 7000 uwag pochodzących z setki organizacji reprezentujących przemysł, uniwersytety, administrację i środowiska wojskowe. Jednocześnie ANSI zdefiniował oprogramowanie służące do uwiarygodnienia kompilatorów ADY, zawierające 1800 testów i 100 tys. linii programu. Niezależnie od prac nad kompilatorami na duże i małe maszyny, przedsiębiorstwo ALSYS (p. W SKRÓCIE — nr 7—8/83) opracowało programy wspomaganego komputerem nauczania ADY.

*

Zawarte ostatnio porozumienie między rządami Stanów Zjednoczonych i Japonii ma służyć szerszemu otwarciu japońskiego rynku. Prawdopodobnie jako pierwszy skorzysta z tych udogodnień znany producent układów scalonych — INTEL. Jak oświadczył na spotkaniu amerykańskiego Stowarzyszenia Przemysłu Półprzewodników, Robert Noyce — prezes INTELA, jego firma omawia z Japonią sprawę wybudowania tam swojej fabryki. Szczegóły tej operacji są ciągle dyskutowane, wiadomo jednak, że wykorzystane zostaną niskoprocentowe preferencyjne kredyty z Japońskiego Banku Rozwoju.

*

Firma MICROSOFT, znany producent oprogramowania, nawiązała współpracę z dziesięcioma firmami japońskimi w celu sformułowania światowego standardu komputera domowego, sprzedawanego w cenie poniżej 200 dol. Takie porozumienie w sposób oczywisty ułatwiłoby rozwój oprogramowania. Standardowym produktem będzie komputer zbudowany przy użyciu procesora Z80 i wyposażony w pamięć o pojemności 64 K bajtów, a także podzbiór 16-bitowego oprogramowania stworzonego przez MICROSOFT dla IBM PC. Złoży się na nie system operacyjny (rozszerzany do wersji dyskowej, o ile taki pojawi się w wyposażeniu), język BASIC, program redagujący zapis na ekranie oraz standardowe połączenie do gier. Inne elementy, które zostaną znormalizowane, to: zbiór znaków, format kasety, klawiatura i połączenie wymiennych modułów ROM. Oczekuje się podjęcia produkcji takich urządzeń jeszcze przed Gwiazdką, by skorzystać z okresu zwiększonego popytu.

*

O popularności mikrokomputerów firmy APPLE świadczy fakt podrabiania ich przez ok. 40 wytwórni azjatyckich. Jak się ocenia, w ubiegłym roku przemycono do Stanów Zjednoczonych co najmniej 1000 sztuk takich nielegalnych kopii, naruszających prawa właściwego producenta. W związku z tym APPLE zwróciła się do Komisji Handlu Międzynarodowego o rozpatrzenie tej sprawy i wydanie nakazu konfiskowania falsyfikatów przez służby celne. Niemniej lepszym zabezpieczeniem będzie w nowym modelu IIe zastosowanie specjalnie zaprojektowanych, nietypowych układów sterujących adresowaniem pamięci i monitorem ekranowym, których podrobienie powinno być nieopłacalne ze względu na koszt i czasochłonność.

*

Ciągły postęp miniaturyzacji pozwolił umieścić mikrokomputer wewnątrz organizmu ludzkiego. Nie chodzi tu oczywiście o ponure wizje dyspozycyjnego społeczeństwa, złożonego z ludzi sterowanych komputerami, ale o nową generację stymulatorów serca. Pierwszy model SEQUICORE II firmy CORDIS (USA) otrzymał właśnie atest komisji dopuszczającej do obrotu sprzęt medyczny. Natomiast w trakcie badań klinicznych znajdują się już następne typy — GEMINI firmy CORDIS oraz COSMOS — z INTERMEDICS. Dzięki zastosowaniu specjalnie do tego celu opracowanych procesorów udało się zmieścić całość urządzenia w obudowie wszczepianej pod skórę pacjenta, zaś technologia CMOS pozwoliła zmniejszyć pobór prądu do 10 μA, co umożliwia pracę przez 6—10 lat bez wymiany baterii. Następną zaletą komputerowych stymulatorów jest ich niespotykana dotąd elastyczność w doborze rytmu serca, zależnie od warunków, gdyż system działa jednocześnie jako czujnik i układ wykonawczy. W szczególności istnieje możliwość jego pracy „dorywczej", tzn. tylko w okresach wykrycia anomalii pulsu. Jednocześnie zastosowanie sterowania komputerowego, a więc programowanego, pozwala lekarzowi na przeprogramowanie urządzenia, stosownie do aktualnego stanu zdrowia pacjenta — służy do tego zewnętrzny, podręczny programator. Dodatkowo, nowa generacja stymulatorów rozszerza dotychczasową diagnostykę; dzięki danym magazynowanym w pamięci RAM lekarz może dowiedzieć się — na przykład — ile razy każda strona serca pracowała samodzielnie, a ile razy musiała być wspomagana. Oczywiście ten typ zastosowań zmusił konstruktorów do położenia szczególnego nacisku na niezawodność, np. COSMOS zawiera mniejszy, zapasowy układ scalony, wkraczający do akcji w razie awarii układu głównego. Musiano także rozwiązać sporo specyficznych problemów technicznych, m.in. wyeliminowanie niebezpiecznego dla życia samosprzężenia urządzenia (tzn. odczytywania wysyłanych przez nie impulsów jako pochodzących od serca). Udało się jednak pokonać te trudności. Zasadniczą wadą tych stymulatorów jest obecnie ich cena — 5000 dol. w porównaniu z 2000 dol. za starsze modele.

Syc. C., Paczocha J.: Sieci teleinformatyczne

INFORMATYKA 1983, nr 9, s. 2

Uzasadnienie potrzeby oraz możliwości realizacji w Polsce ogólnodostępnej sieci teleinformatycznej. Dokonano analizy rozwiązań technicznych dla racjonalnego wyboru rodzaju sieci w aktualnej sytuacji krajowej.

Сых С., Пачоха Й.: Телеобрабатывающие сети

INFORMATYKA 1983, № 9, стр. 2

Обоснование потребности и возможности реализации в Польше общедоступной сети телеобработки. Совершён анализ технических решений для рационального выбора вида сети в настоящей отечественной обстановке.

Bartol W.: Programowanie za pomocą współprogramów

INFORMATYKA 1983, nr 9, s. 7

Charakterystyka własności współprogramów jako narzędzi ułatwiających pisanie programów o strukturze modularnej. Własności te zilustrowano przykładami zapisanymi w notacji języka LOGLAN.

Бартоль В.: Программирование с помощью сопрограмм

INFORMATYKA 1983, № 9, стр. 7

Характеристика свойств сопрограмм как орудий, облегчающих писание программ модульной структуры. Эти свойства иллюстрируются примерами, записанными с использованием нотации языка LOGLAN.

Owczarczyk J.: Systemy graficzne: percepcja wzrokowa

INFORMATYKA 1983, nr 9, s. 10

Teoretyczne podstawy najważniejszych aspektów percepcji wzrokowej użytkownika systemu graficznego. Właściwe zrozumienie i wykorzystanie psychofizjologicznych właściwości systemu wzrokowego człowieka warunkuje zwiększenie skuteczności odbioru komputerowej informacji graficznej.

Овчарчык Й.: Графические системы: Зрительное восприятие

INFORMATYKA 1983, № 9, стр. 10

Теоретические основы наиболее важных аспектов зрительного восприятия. Правильное понимание и использование психофизиологических свойств зрительной системы человека обуславливает увеличение эффективности восприятия машинной графической информации.

Wilczek T.: Aspekty użytkowe języków symulacyjnych (I). Symulacja dyskretna

INFORMATYKA 1983, nr 9, s. 13

Klasyfikacja, przegląd oraz ocena wartości użytkowej stosowanych w kraju języków i pakietów symulacyjnych. Omówiono języki i pakiety przeznaczone do symulacji sysmów dyskretnych.

Вильчек Т.: Эксплуатационные аспекты языков моделирования (I). Дискретное моделирование

INFORMATYKA 1983, № 9, стр. 13

Классификация, обзор и оценка потребительной стоимости применяемых в стране языков и пакетов моделирования. Обсуждены языки и пакеты, предназначенные для моделирования дискретных систем.

Grabowski J.: Przegląd mikroprocesorów 16-bitowych (6). NATIONAL SEMICONDUCTOR 16000

INFORMATYKA 1983, nr 9, s. 17

Szósta część charakterystyki najbardziej popularnych mikroprocesorów 16-bitowych, obejmująca omówienie układów NATIONAL SEMICONDUCTOR 16000.

Грабовски Й.: Обзор 16-битовых микропроцессоров (6). NATIONAL SEMICONDUCTOR 16000

INFORMATYKA 1983, № 9, стр. 17

Шестая часть характеристики наиболее популярных 16-битовых микропроцессоров, содержащая обсуждение схем NATIONAL SEMICONDUCTOR 16000.

Karczmarczuk J.: Jeszcze o ZX81

INFORMATYKA 1983, nr 9, s. 22

Rozszerzenie charakterystyki technicznej i użytkowej komputera osobistego ZX81 w oparciu o doświadczenia użytkownika. Omówiono praktyczne możliwości programowania w języku BASIC oraz realizacji rozbudowy sprzętowej.

Карчмарчук И.: Ещё о ZX81

INFORMATYKA 1983, № 9, стр. 22

Расширение технической и эксплуатационной характеристи персонального компьютера ZX81 на базе опыта пользователя. Обсуждены практические возможности программирования на языке BASIC и выполнение аппаратного расширения.

Syc. C., Paczocha J.: Teleprocessing networks

INFORMATYKA 1983, No. 9, p. 2

Justification of the need and realization possibility of public teleprocessing network in Poland. Analysis of technologic solutions for rational choice of network type in actual Poland's situation is presented.

Syc C., Paczocha J.: Datenfernverarbeitungsnetze

INFORMATYKA 1983, Nr. 9, S. 2

Eine Begründung des Bedarfes und der Realisationsmöglichkeiten eines öffentlichen Datenfernverarbeitungsnetzes in Polen. Es wurde eine Analyse der technischen Lösungen für eine rationelle Wahl der Netzart bei aktueller Lage des Landes angegeben.

Bartol W.: Programming with coroutines

INFORMATYKA 1983, No. 9, p. 7

Property characteristics of coroutines, which are used as tool for writing simplification of programs with modular structure. The property was illustrated with examples written in LOGLAN notation.

Bartol W.: Programmierung mit Hilfe von Mitprogrammen

INFORMATYKA 1983, Nr. 9, S. 7

Eine Charakteristik der Eigenschaften von Mitprogrammen, die als Hilfsmittel das Schreiben der Programme mit Modularstruktur erleichtern. Die Eigenschaften wurden mit Beispielen in LOGLAN-Notation illustriert.

Owczarczyk J.: Graphics systems: visual perception

INFORMATYKA 1983, No. 9, p. 10

Theoretic foundations of main aspects of visual perception by graphics system user. Proper understanding and using psychophisiologic facilities of human visual system gives conditions for extending efficiency of computer graphics information receipt.

Owczarczyk J.: Graphische Rechnersysteme: Gesichtsperzeption

INFORMATYKA 1983, Nr. 9, S. 10

Theoretische Grundlagen der wichtigsten Aspekte der Gesichtsperzeption eines Benutzers von graphischem System. Richtiges Verständnis und Ausnutzung der psychophysiologischen Eigenschaften des menschlichen Gesichtssystems bedingt eine Wirksamkeitsteigerung der graphischen Rechnerinformation.

Wilczek T.: Utility aspects of simulation languages (1). Discrete simulation

INFORMATYKA 1983, No. 9, p. 13

Classification, preview and evaluation of utility value of home applied simulation languages and packages. Languages and packages for discrete systems simulation are discussed.

Wilczek T.: Nutzaspekte der Simulationssprachen (1). Diskrete Simulation

INFORMATYKA 1983, Nr. 9, S. 13

Klassifikation, Übersicht und Beurteilung des Nutzwertes der in Polen angewendeten Simulationssprachen und — Pakete. Es wurden Sprachen und Pakete zur Simulation der diskreten Systeme besprochen.

Grabowski J.: Survey of 16-bit microprocessors (6). NATIONAL SEMICONDUCTOR 16000

INFORMATYKA 1983, No. 9, p. 17

Sixth part of most popular 16-bit microprocessor characteristics — presentation of the NATIONAL SEMICONDUCTOR 16000.

Grabowski J.: Vorschau von 16-Bits-Mikroprozessoren (6). NATIONAL SEMICONDUCTOR 16000

INFORMATYKA 1983, Nr. 9, S. 17

Sechster Teil der Charakteristik von populärsten 16-Bits-Mikroprozessoren, die eine Besprechung der Schaltkreise von NATIONAL SEMICONDUCTOR 16000 umfasst.

Karczmarczuk J.: More about ZX81

INFORMATYKA 1983, No. 9, p. 22

Extension of technologic and utility characteristics of ZX81 personal computer based on user's experience. Practical possibilities for BASIC programming and hardware extension are discussed.

Karczmarczuk J.: Noch über ZX81

INFORMATYKA 1983, Nr. 9, S. 22

Eine Erweiterung der technischen und verwendbaren Charakteristik von ZX81 Personalrechner auf Grund der Benutzerserfahrungen. Es wurden praktischen Möglichkeiten des Programmierens in BASIC, sowie der Hardwareerweiterung, besprochen.

Rynek dla informatyki

Informatykę zawodową można organizować w różnym zakresie. Można jej działaniem objąć zebranie koniecznych informacji, spisanie ich, przetworzenie — aż do dystrybucji wyników. Można też wszakże zawęzić jej pracę do mechanicznego przechowania i przetworzenia informacji, bez wnikania w meritum sprawy. Ale wówczas sprowadzamy własne usługi jedynie do zaplecza technicznego.

Polem działania informatyki jest właściwie każda dziedzina ludzkiej aktywności. Wszędzie mamy informacje i wszędzie występuje jej przetwarzanie. Różnorodność problemów, przed jakimi zatem stoją informatycy, wymaga sporej inteligencji. A niestety, inteligentne zachowanie, to zachowanie adekwatne do sytuacji. Nie daje się ono pogodzić z ustabilizowanymi formami systemu hierarchicznego, którego organiczną cechą jest niezdolność przystosowywania się do zmiennych warunków. System hierarchiczny jest ze swej istoty mało inteligentny.

Wniosek może być jeden. Albo malejący liczebnie zawodowi informatycy, obdarzeni przecież zdolnościami logicznego myślenia, wyłamią się z ram aktualnego systemu organizacyjnego, albo nadal nie będziemy mieli zawodowej informatyki w kraju. Nic tu nie pomoże efektowny protest, potrzebna jest precyzyjna robota organizacyjna, od podstaw. Właśnie informatyka musi wdrożyć nie stosowane w Polsce elastyczne, wielowymiarowe formy organizowania. W przeciwnym razie sztywny, niesprawny system organizacyjny (i jego kierownictwo) — nie dopuści do jej rozwoju.

Potrzebne są różnorodne, dostosowane do potrzeb organizacje. Potrzebni są indywidualni informatycy, drobne zespoły o charakterze spółdzielczym, drobne przedsiębiorstwa państwowe, zespoły konsultantów i doradców przy towarzystwach wyższej użyteczności publicznej i uczelniach. Potrzebne są także duże jednostki gromadzące pełne inicjatywy zespoły wokół silnej bazy technicznej.

Postęp techniczny i technologiczny w informatyce światowej jest szczególnie szybki. Nie lubimy k o n k u r e n c j i — to słowo pochodzi z innego systemu. W s p ó ł z a w o d n i c t w o sami sobie obrzydziliśmy. Przyjmijmy, że chodzi o g r ę. Bez tej gry, która tworzy zagrożenie tym, co nie nadążają, nie ma i nie będzie postępu. Po co się wysilać, jeśli można bezkarnie trwać w bezruchu. Bezpieczniej i mniej wysiłku. Interes, ambicja, zagrożenie — to podstawowe motory postępu. Kto nie trafi w potrzeby, albo źle obliczy skalę — zginie. W jego miejsce wejdą inni, lepsi. A sumaryczny efekt musi być wtedy wysoki. Wiadomo, miarą efektu jest zaspokojenie potrzeb wyrażone przez sprzedaż usług, a nie — produkcja usług według anonimowo wymyślonych planów i zadań.

Przy ogromnych brakach na rynku informacji, forsuje się tezę o konieczności zwężenia pola działania informatyków do usługowego, technicznego przetwarzania danych. Sieci ZETO, ETOB i inne — pełne sfrustrowanych programistów, konserwatorów, technologów, operatorów — tkwią w marazmie. Obok — „morze" potrzeb, ale sięgnięcie doń bez radykalnej zmiany organizacji nie jest możliwe.

Z doświadczeń wynika, że przy pełnej obsłudze informacyjnej klienta, zaledwie ok. 30% stanowią techniczne usługi informatyczne i to łącznie z rozwojem systemów przetwarzania danych. 70% to prace pozostałe — począwszy od doradztwa organizacyjnego i informacyjnego, na organizowaniu obiegu informacji skończywszy.

Chcemy dopracować się rynku w naszym kraju. Dla informatyka jest sprawą oczywistą, że istnienie dostawców i odbiorców towarów i usług jeszcze rynku nie tworzy. Dopiero tkanka informacyjna łącząca jednostki obecne na rynku umożliwia ich współdziałanie. Bez informacji niemożliwy jest wybór, a nawet — znalezienie partnera. Dopiero marketing, łączący potrzeby z możliwościami ich zaspokojenia, daje szansę społecznej efektywności wytworzonych dóbr i usług. Informatyka nadal czeka w tej sferze na wielkie zlecenia. Ale przecież w odpowiednio małej skali można by już chyba ryzykować...

Nowa sytuacja ekonomiczna przedsiębiorstw z trudem przebija się do świadomości ich kierownictw. Ekonomika w okresie inflacji różni się od ekonomiki czasu stabilizacji. Z obserwacji wynika, że często przedsiębiorstwa są bezradne. Bez automatycznego modelowania zmieniającej się jak w kalejdoskopie sytuacji, a nawet bez automatycznej kalkulacji, nie można prowadzić prawidłowej działalności. Sytuacja ekonomiczna przedsiębiorstw staje się zasadniczo niezgodna z ich sytuacją finansową!

Brak odpowiednich usług informatycznych wprost zagraża reformie. Niedostateczna moc przetwarzania danych może zmusić do ponownego zastosowania prymitywnych metod rozliczeniowych w miejsce rozrachunku ekonomicznego. Brak możliwości modelowania prowadzi do paradoksów jak niżej opisany. Sprawa brzmi jak żart, niestety odpowiada rzeczywistej analizie.

Przedsiębiorstwo chciało odzyskać część materiałów budowlanych. Praca potrzebna dla odzysku miała kosztować 3 mln zł rocznie. Wartość odzysku materiałów szacowano na 10 mln zł. Wspaniały interes! Oszczędzone materiały i jeszcze taki zysk! Ale, niestety, produkcja wprawdzie nie rośnie, ale rośnie fundusz płac i zysk. Na FAZ trzeba by wpłacić 4,5 mln zł oraz podatek dochodowy 4 mln. W sumie przedsiębiorstwo musiałoby dopłacić do interesu 1,5 mln zł. Żegnajcie materiały — trzeba je tylko sprytnie zakopać. Nawet ulgi w opłatach nic tu nie pomogą.

Na koniec kilka refleksji o organizacji, również w informatyce. Organizujemy się, aby ułatwić, uprościć, poprawić sobie życie. Mnie, tobie, nam łatwiej bronić się, pracować, współdziałać, jeżeli jesteśmy zorganizowani. Każdy robi to, co mu stosunkowo robić najłatwiej i pewnie najprzyjemniej i za to otrzymuje coś innego. Organizacja jest rzeczą ludzką i łatwo odróżnić organizację dobrą od zlej. Może też być inaczej: organizowani jesteśmy na rzecz gospodarki narodowej, społeczeństwa, państwa.

Jeżeli założymy, że organizacje są dla ludzi, działalność produkcyjną i usługową będziemy kształtować w celu wymiany. Dążyć będziemy do tworzenia organizmów niezawodnie spełniających usługi z maksymalnym efektem wymiany. Struktury będą elastyczne, wykorzystujące maksimum inicjatywy i nadarzających się okazji, nastawione na obiekt wymiany — klienta. Miarą oceny będzie odpowiedzialność za osiągnięty wynik — efekt wymiany.

Jeżeli tworzymy organizację dla służby, musimy tworzyć struktury dokładnie rozdzielone, z dokładnym opisem czynności, umożliwiające jednoznaczne ustalenie winnego. Kryterium oceny staje się skrupulatne przestrzeganie przepisów i poleceń przełożonych.

W systemie nakazowo-rozdzielczym, dostosowanym do służby, już z samej nazwy wynika posłuszeństwo. A w systemie zreformowanym niezbędna zdaje się być gospodarność.

Powstaje dylemat. Czy utrzymanie obecnej sztywnej struktury organizacyjnej informatyki, praktyczne zablokowanie konkurencji, jest dowodem sprawności czy niesprawności kierownictwa. Jeżeli w nowej sytuacji społeczno-gospodarczej pojawią się mechanizmy ekonomiczno-rynkowe, będzie to niesprawność kierownictwa, które spowoduje co najmniej utratę szans informatyki. Jeżeli zaś odwrotnie — system społeczno-gospodarczy będzie rozwijał się w kierunku zdyscyplinowanej realizacji centralnie ustalonych planów dyrektywnych, będzie to objaw mądrego przewidywania.

Czy na długo? Wątpię. Inaczej należałoby przyjąć, że możliwe jest pojawienie się rozwiniętej gospodarki i społeczeństwa socjalistycznego w trybach machiny centralizmu biurokratycznego.

ANDRZEJ ZIENKIEWICZ

Zasady prenumeraty

Zamówienia i przedpłaty na prenumeratę INFORMATYKI przyjmuje Zakład Kolportażu Wydawnictwa NOT SIGMA. Adres pocztowy: Wydawnictwo NOT SIGMA — Zakład Kolportażu, 00-950 Warszawa, skr. poczt. 1004. Konto bankowe: 1036-7490-139-11, III O/M NBP w Warszawie.

Jednostki gospodarki uspołecznionej, instytucje i organizacje przesyłają zamówienia (w 1 egz.) zawierające: tytuł czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty i pełny adres zamawiającego z kodem pocztowym, a także numer konta bankowego zamawiającego oraz (ewentualnie) adresy odbiorców, którzy na zlecenie zamawiającego mają prasę otrzymywać.

Dopisując w zamówieniu PRENUMERATA STAŁA, zamawiający nie będzie musiał corocznie ponawiać zamówienia, a jedynie dokonywać przedpłaty według aktualnie obowiązujących cen.

Warunkiem realizacji zamówienia jest równoczesne dokonanie odpowiedniej wpłaty na ww. konto Wydawnictwa NOT SIGMA.

Prenumeratorzy indywidualni dokonują wpłaty przekazem na ww. konto, podając na odwrocie odcinka dla adresata-posiadacza rachunku: tytuł czasopisma, liczbę zamawianych egzemplarzy oraz okres prenumeraty.

Zamówienia i wpłaty przyjmowane są w terminach:

- do 15 listopada — na I kwartał, I półrocze i cały rok
- do 28 lutego — na II kwartał
- do 31 maja — na III kwartał i II półrocze
- do 31 sierpnia — na IV kwartał.

Uwaga: przy podawaniu kodu pocztowego i numeru konta bankowego obowiązuje bardzo czytelne pismo.

Prenumerata kwartalna — 225 zł, półroczna — 450 zł, roczna — 900 zł. Prenumerata ze zleceniem wysyłki za granicę jest dwukrotnie droższa.

Od 1 lipca br. wprowadzona została prenumerata ulgowa dla członków stowarzyszeń naukowo-technicznych NOT, studentów wyższych uczelni, uczniów szkół zawodowych. Warunkiem jej uzyskania jest poświadczenie blankietu dla nabywcy indywidualnego (na odcinku dla adresata) przez właściwe stowarzyszenie NOT, wyższą uczelnię lub szkołę zawodową. Cena prenumeraty ulgowej: kwartalnie — 150 zł, półrocznie — 300 zł, rocznie — 600 zł.

Dodatkowych informacji o prenumeracie udziela: Zakład Kolportażu, tel. 40-00-21 w. 293, 295, 217 oraz 40-35-89. Egzemplarze archiwalne można nabywać w Klubie Prasy i Informacji Technicznej w Warszawie, ul. Mazowiecka 12, tel. 27-43-65.

W numerze 10/83 przedstawimy:

Maliszewski K.: Przemysłowy FORTRAN czasu rzeczywistego. Część 1
Liszyński Z.: Programowanie procesów współbieżnych (2).
Środowisko programowe
Majewski J. T., Wagner K.: Język SIMBOL dla MERY 400
Wilczek T.: Aspekty użytkowe języków symulacyjnych (2).
Symulacja ciągła i ciągło-dyskretna

W najbliższych numerach:

- Historia systemów baz danych
- Język ADA w testach
- Porównanie mikroprocesorów 16-bitowych
- Interakcyjny system przetwarzania obrazów
- SART — automatyczne redagowanie tekstów